
INPUT
Publicación práctica para usuarios de MSX
Revista mensual 1986
Año 1 - Número 7 Precio 350 Ptas.
MSX
ZEN; el maestro ensamblador
Calcula tus préstamos y capitalizaciones
INTELIGENCIA ARTIFICIAL: el ordenador juega a las cuatro en raya
Los modos de textos del MSX



LAS TRES LUCES DE GLAURUNG

UN PROGRAMA HECHO EN ESPAÑA QUE ESTA SORPRENDIENDO EN EUROPA

Un guerrero va a enfrentarse, sólo, a los incontables peligros que acechan en el Castillo bajo la Montaña, más allá de donde alcanza la luz del Sol y de donde se atreven a llegar los corazones más valerosos.

La fuerza, la astucia y la habilidad, van a medirse con el hierro, el fuego y la hechicería, en uno de los más tremendos choques entre el Bien y el Mal que el Universo ha presenciado jamás.

ERBE

DISTRIBUIDOR EXCLUSIVO PARA ESPAÑA
ERBE SOFTWARE.
SANTA ENGRACIA, 17. Tel: 447 34 10
DELEGACION BARCELONA.
Avd. MISTRAL, 10. Tel. (93) 432 07 31

AÑO 1 NUMERO 7



DIRECTOR:
Alejandro Diges
DIRECTOR TECNICO:
Roberto Menéndez
COORDINADOR EDITORIAL:
Francisco de Molina
DISEÑO GRAFICO:
Tomás López
COLABORADORES:
Antonio Taratiel, Luis R. Palencia, Francisco Tórtola, J. A. Feberero, Esther de la Cal, Ernesto del Valle, Equipo Molisoft, Javier Portillo.

INPUT MSX es una publicación de PLANETA-DE AGOSTINI, S. A.

GERENTE DIVISION DE REVISTAS:
Angel Sabat

PUBLICIDAD: José Real-Grupo Jota
Madrid: c/ Gral. Varela, 35, 3.º-11
Teléf. 270 47 02/03
Barcelona: Avda. de Sarriá, 11-13, 1.º
Teléf. 250 23 99

FOTOMECANICA: Ochoa, S. A.

COMPOSICION: EFCA, S. A.

IMPRESION: Sirven Grafic
C/ Gran Vía, 754-756. 08013 Barcelona
Depósito legal: B-21953-1986

SUSCRIPCIONES: EDISA,
López de Hoyos, 141. 28002 Madrid
Teléf. (91) 415 97 12

REDACCION:
Paseo de la Castellana, 93, 14.ª
28046 Madrid. Teléf. 456 54 13

DISTRIBUIDORA
R.B.A. PROMOTORA DE EDICIONES, S. A.
Travesera de Gracia, 56. Edificio Odiseus.
08006 Barcelona.

El precio será el mismo para Canarias que para la Península y en él irá incluida la sobretasa aérea.

INPUT MSX es una publicación controlada por



INPUT MSX

SUMARIO

NO TODO SON Ks

A estas alturas del año, cuando sólo nos resta por ver lo que pueda ofrecernos el **SIMO,** y ya se intuye la presión publicitaria de las campañas de Navidad, es el momento de hacer un repaso de lo que microinformáticamente ha sido el año 86.

Tanto **MSX** como **Commodore** y **Sinclair** parecían dispuestos a satisfacer la demanda de Ks de memoria que exigían los usuarios para sentir que sus máquinas no se quedaban atrás en el imparable avance de los micros por alcanzar las capacidades mitológicas de los grandes ordenadores. Sin embargo, en estos últimos meses, los fabricantes han decidido reconsiderar el camino que habrían de tomar sus nuevos productos.

Los que intuían una progresión geométrica de las capacidades de memoria, que se proyectaba así hacia el infinito, se han encontrado con que **Commodore** ha diseñado una bonita carcasa para su clásico **C-64.** Y que **Sinclair-Amstrad,** aunque ha hecho bastante más por su 128 (le ha incorporado un *cassette,* un *chip* de sonido y alguna cosa más) no ha superado el listón de los 128 K alcanzado el invierno pasado ni en un solo bit. Tal vez el contrapunto a esta situación lo ponga la gran familia **MSX.** Su *hard* se ha desarrollado dentro de las previsiones, y según qué modelos, los micros de la segunda generación tienen memorias que están entre los 192 y los 384 Ks; la manipulación de vídeo es una realidad y las cualidades gráficas son sensacionales. Sin embargo tampoco así hemos podido lograr toda la dicha. Aunque todas estas capacidades están ahí, falta el *soft* capaz de ponerlas de manifiesto. De momento tendremos que conformarnos con los programas desarrollados para la primera generación y las adaptaciones de programas de otras máquinas.

LOS MEJORES DE INPUT

Hemos pensado que es interesante disponer de un ***ranking*** que ponga en claro, mes a mes, cuáles son los programas preferidos de nuestros lectores. Para ello, es obligado preguntaros directamente y tener así el mejor termómetro para conocer vuestras preferencias. Podéis votar por cualquier programa aunque no haya sido comentado todavía en **INPUT**.

El resultado de las votaciones será publicado en cada número de **INPUT**.

Entre los votantes sortearemos 10 cintas de los títulos que pidáis en vuestros cupones.

Nota: No es preciso que cortéis la revista, una copia hecha a máquina o una simple fotocopia sirven.

Enviad vuestros votos a: **LOS MEJORES DE INPUT** P.º de la Castellana, 93. Planta 14. 28046 Madrid

ELIGE TUS PROGRAMAS

Primer título elegido ____

Segundo título elegido ____

Tercer título elegido ____

Programa que te gustaría conseguir ____

Qué ordenador tienes ____

Nombre ____

1.er Apellido ____

2.º Apellido ____

Fecha de nacimiento ____

Teléfono ____

Dirección ____

Localidad ____

Provincia ____

INPUT MSX N.º 7

MSX2; UNO MAS EN LA FAMILIA

Sony ha decidido aumentar la familia de equipos de la segunda generación con un recién nacido, de inmejorable aspecto, que ha sido bautizado como: HBF 700F.

El aspecto externo de la máquina es idéntico al del anterior modelo segunda generación **HB-F500P**, pero las prestaciones son muy superiores.

La memoria RAM alcanza nuevas cotas y se sitúa en un total de 384 Kbytes, 128K dedicados al procesador de vídeo y el resto, un total de 256K, destinados a los programas del usuario. La unidad de diskettes incorporada ofrece una capacidad de 720 Kbytes, en formato de doble cara doble densidad.

Quiza lo más interesante sea el software incluido en diskette, formado por un conjunto de 5 programas: HI-BRID, una especie de GEM o interface interactivo de ventanas e iconos, HI-TEXT para el tratamiento de textos, HI-BASE para manejar archivos, HI-CALC en funciones de hoja electrónica y HI-GRAPH como especialista en gráficos.

Todos estos programas se manejan a través del ratón que, por primera vez, viene incluido con el equipo. La oferta no puede ser más atractiva.

****

STRIP POKER MSX

Serma ha previsto para este mes el lanzamiento, en versión **MSX**, del popular "strip poker" de **Samantha Fox**, que tantos éxitos ha cosechado en sus versiones para Sinclair y Commodore. Por otro lado nos han llegado noticias de la firma holandesa **Aackosoft** que ha lanzado, bajo el título de **Red** Lights of Amsterdam (Las luces rojas de Amsterdam), un programa sobre el mismo tema, para **MSX2**, en el que se hace uso de síntesis vocal y de imágenes digitalizadas. Sin embargo, hasta el momento, no hay noticias sobre la posible importación de este soft.

****

DISCO DURO PARA EL X'PRESS

Spectravideo presentó, durante el último **Sonimag**, una serie de interesantes novedades. La más "salvaje" de todas fue la unidad de disco duro de 10 Mbytes conectable al **SVI** 738 X'Press. Esta unidad esta prevista para trabajar bajo el sistema operativo CP/M y saldrá al mercado posiblemente a partir del mes de noviembre. Con los 10 Megas y la velocidad del disco duro, con las características del **X'Press** (próximas a las de los segunda generación) y con el soft profesional que comercializa **Spectravideo** (dBASE II, Multiplan, etc) se puede empezar a pensar en trabajos serios desde el estandar **MSX.**

Aparte de este Hard-Disk, pronto podremos disfrutar de un sintetizador de voz para **MSX**, que "dirá" a través del altavoz del televisor o del monitor cualquier frase que se nos ocurra y que previamente hayamos tecleado. El software del sintetizador, que lleva el nombre de **Charlatán**, se ofrecerá en cassette y diskette.

La última novedad de la casa es un joystick de "carreras"; el **Quickshot** II Turbo. Aparte de su nuevo aspecto, incorpora microinterruptores en lugar de los clásicos contactos. Con ello se consigue una mejor repuesta.

****

CD ROM

Sony y la firma americana **Knowledgeset**, dedicada a la edicion de soft, han llegado a un acuerdo sobre un soft de gestión de CD ROM. El producto, de nombre Knowledge Retrivial System, permitirá la gestión de un CD ROM por parte del ordenador que tendrá acceso a 600 Mbytes de memoria de lectura.

****

PRESTAMOS Y CAPITALIZACIONES

Si alguien aún no ha pedido un préstamo a un Banco, que cruce los dedos o toque madera, pero sin perder la esperanza... Para cuando llegue ese momento, el programa que vamos a considerar será de gran ayuda.

El programa vale también para aquellos que deseen disponer de un cierto capital en una fecha determinada como puede ser el caso de la jubilación o ahorrar lo necesario para dar la entrada exigida en la compra de un piso.

Tanto en uno como en otro caso es necesario hacer muchas cábalas y tanteos, jugando con meses, número de plazos, intereses, etc. antes de tomar una decisión definitva. El ideal sería tener todo esto bien estudiado tranquilamente en casa y llevarnos la papeleta preparada cuando acudamos al banco o a la Caja de Ahorros.

En el programa PRESTAMOS Y CAPITALIZACIONES hemos considerado sólo aquellos casos de uso más frecuente, pero teniendo en cuenta que podemos hacer un estudio completo en unos pocos segundos a partir de ciertos parámetros, no nos será difícil tantear cualquier situación por enrevesada que parezca. Lo que si hay que aclarar es que el programa no tiene en cuenta los impuestos o retenciones complementarias que hacen los bancos o el gobierno, como puede ser el **IVA**, comisiones, pólizas, etc.

El programa es muy fácil de manejar a través de su Menú:

1 - TOMA DE DATOS (CUOTA FIJA)

2 - VISUALIZACION TABLA DE PRESTAMOS

3 - IMPRESION TABLA DE PRESTAMOS

4 - VISUALIZACION TABLA DE CAPITALIZACIONES

5 - IMPRESION TABLA DE CAPITALIZACIONES

6 - TOMA DE DATOS (CUOTA VARIABLE)

7 - VISUALIZACION TABLA CAPITALIZACIONES

8 - IMPRESION TABLA DE CAPITALIZACIONES

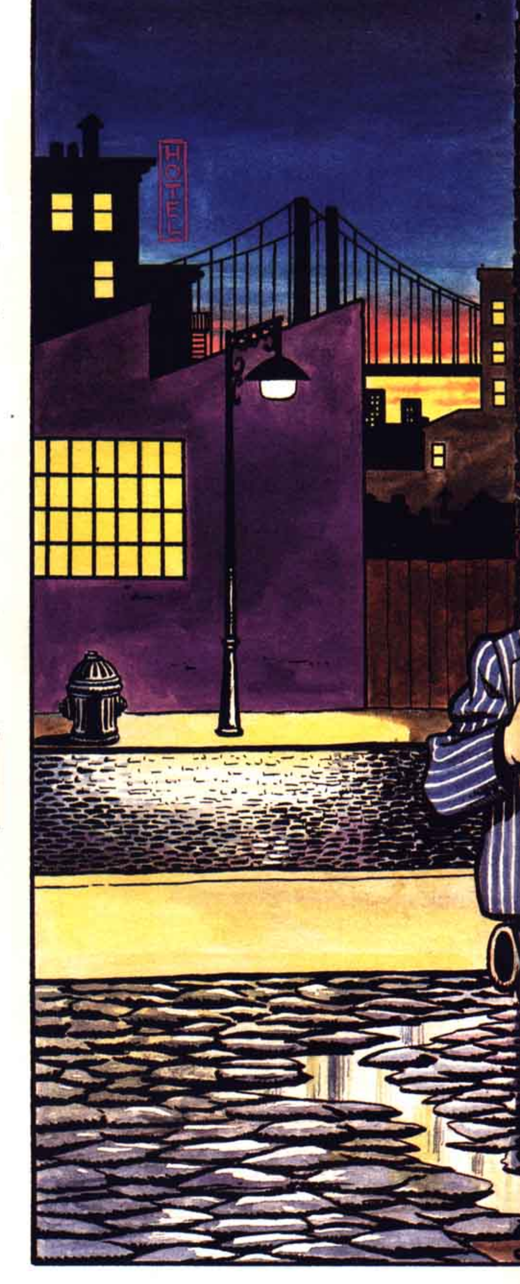
Se han considerado dos situaciones: (1 a 5) para cuotas de amortización o capitalización constantes o (6 a 8) para porcentaje preestablecido. Con ello, tal como se indicaba anteriormente se resuelven la mayoría de los casos prácticos de una forma directa y el resto se puede resolver mediante unos tanteos muy sencillos.

AMORTIZACION DE PRESTAMOS O DEUDAS

Cuando acudimos a un banco o Caja de Ahorros a solicitar un préstamo deberemos fijar la cantidad deseada, así como el tiempo en que pretendemos devolverlo y el número de plazos dentro de cada año. Por ejemplo 600.000 ptas. en 5 años con pagos trimestrales.

Lo normal es que los pagos sean todos iguales y por ello, este es el caso contemplado en el programa. Aún cuando todos los pagos sean iguales, en cada uno de ellos hay una cantidad que corresponde al interés que se lleva el banco o entidad de crédito por prestarnos el dinero y la restante corresponde a la parte de capital que amortizamos cada vez. En cada plazo se paga menos intereses que en el anterior y por tanto se amortiza una mayor proporción.

El banco nos fija la cuota a pagar en función del tanto por ciento del interés que nos aplique. En nuestro caso hemos considerado que el porcentaje no varía a lo largo de todo el período de amortización. Aún cuando el pago pudiera ser mensual, cuando se habla del % del interés, salvo que se indique otra cosa, se refiere a un año.



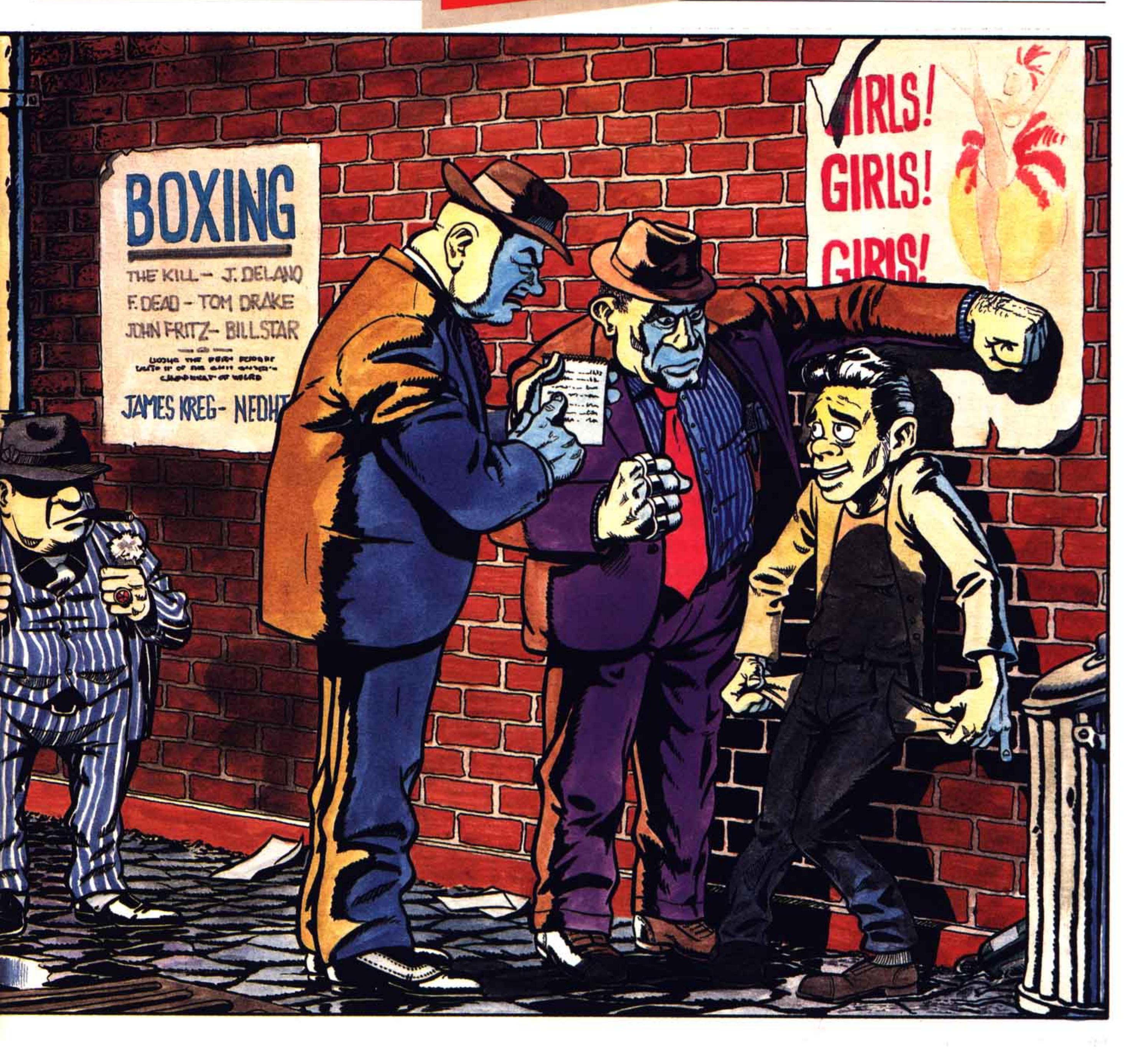
En el primer apartado del menú (TOMA DE DATOS) se nos preguntan todos los datos necesarios:

- CAPITAL DESEADO
- INTERES ANUAL (%)
- NUMERO DE PLAZOS
- MESES POR PLAZO

El programa nos dá la oportunidad de rectificar o reconsiderar los datos:

DESEA MODIFICAR? S/N (para no modificar basta pulsar 'N', 'n' o 'RETURN')

Como los datos valen lo mismo para



definir un préstamo que una capitalización, como veremos más adelante, el programa nos pide que precisemos la operación:

PRESTAMO / CAPITALIZ. (P/C)?

El ordenador nos calcula e imprime en pantalla el valor de la cuota fija que deberemos pagar así como el total de intereses que habremos pagado una vez amortizado todo el capital prestado (la verdad es que normalmente sólo servirá para tirarnos de los pelos o para forzarnos a acordarnos de los bancos de una manera poco amable...). Por ejemplo si solicitamos un millón de pesetas a devolver en pagos mensuales durante cinco años y el banco nos pide un 18% de interés anual deberemos pagar cada més 25.393 ptas. y al final habremos pagado el millón que solicitamos más 523.580 ptas.

Si hubieramos elegido pagar una vez al año en lugar de cada mes, cada pago sería de 319.778 ptas. con un costo de intereses de 598.890 ptas.

Quizá nos demos cuenta que no podemos pagar tanto al més y decidamos tantear pedir menos o pagar a más largo plazo. Basta con volver a introducir los nuevos datos en la opción 1 y en unos segundos tendremos la respuesta.

La situación es análoga si queremos calcular los detalles de una deuda en la que el acreedor acepta financiarla, tal como sería el caso de cualquier «compra a plazos» (casa, coche, electrodomésticos, etc.).

Con la opción 2 podemos visualizar en pantalla el desglose de cada una de

las cuotas, así como la deuda que nos queda por amortizar. Por simplicidad, el programa realiza un redondeo automático a la peseta y además hace un reajuste en los intereses del último pago de forma que la amortización sea perfecta (se produciría una ligerísimo desajuste debido al redondeo que no añadiría mayor exactitud y produciría una efecto no deseado).

Para un uso posterior quizá nos interese conservar impresos los detalles del cálculo e incluso confeccionar tablas con valores típicos para el caso en que no podamos disponer del ordenador en un cierto momento. Para ello existe la opción 3 que es una combinación de las dos opciones anteriores.

Otra forma de amortización sería devolver el capital en parte iguales y pagar en cada período esta cantidad más los intereses devengados por la deuda pendiente. La cuota sería decreciente cada período. El programa no está expresamente preparado para ello por considerar es de menor uso pero no obstante se puede calcular cada plazo con ayuda de la opción 1.

FORMACION DE CAPITALES

La capitalización es un proceso inverso al de amortización de una deuda. En esencia se trata de «prestar a un banco o entidad financiera» un dinero, normalmente en forma periódica, reinvirtiendo todos o parte de los intereses devengados de forma que se genere un aumento del capital con un efecto de «bola de nieve».

Por ser de uso corriente, el programa contempla la posibilidad de capitalización a base de «cuota fija» o de «cuota creciente». Esta segunda forma se suele utilizar en las capitalizaciones a largo plazo (seguro de vida o jubilación) a fin de compensar los efectos de la inflacción. En nuestro caso se considera que el interés del capital no varía durante todo el proceso y que el incremento de la cuota, cuando se elija esta fórmula, se produce de acuerdo a un % establecido.

A. Capitalización con cuota fija

La toma de datos (opción 1) es común a la ya mencionada de amortizacion y al final nos proporciona el valor de la cuota para conseguir el capital estipulado.

Las opciones 4 y 5 nos permiten reflejar en pantalla y en impresora los detalles de cada período.

Ej. Se desea disponer de 10.000.000 de ptas. a los 15 años haciendo aportaciones mensuales iguales.

El banco nos ofrece el 7% de interés.

De la opción 1 deducimos que la cuota debe ser de 31.367 ptas.

B. Capitalizaciones con cuota variable

Las opciones 6, 7 y 8 hacen las funciones equivalentes a 1, 4 y 5 pero con la variante de que se parte de una cuota inicial incrementada anualmente se-

gún un % establecido y se obtiene el capital alcanzado al final del período fijado. Si se desea finalizar con un capital determinado aproximado deberemos hacer una serie de tanteos.

El ordenador nos solicita:
CUOTA INICIAL MENSUAL
INCREMENTO ANUAL %
NUMERO DE PLAZOS
INTERES DEL CAPITAL %

A continuación nos calcula: CAPITAL ALCANZADO (PTAS.)

Ej. Se desea comenzar una capitalización a 15 años con aportaciones mensuales comenzando con 25.000 ptas. e incrementándolas un 8% cada año a fin de combatir la inflación y considerando que cada vez se puede hacer un poco más de esfuerzo económico.

El banco nos garantiza un interés del 6.5%

Al final conseguiremos: 12.594.225 ptas.

Ej. Se desea disponer de 10.000.000 de ptas., aproximadamente a los 10 años haciendo aportaciones mensuales que deseamos aumentar el 10% anual de forma automática.

El banco nos ofrece el 7% de interés.

De la opción 1 deducimos que la cuota inicial puede ser de: 38.100 ptas. para conseguir 10.010.385 ptas. o 38.050 para alcanzar 9.997.248 ptas.

TABLA DE PRESTAMOS

PRESTAMO: 1000000 PTS. AL: 18 % DE INTERES

EN PAGOS DE 109666 PTS. CADA 3 MES(ES)
DURANTE 12 PLAZOS

PLAZO	PARTE CAPITAL	PARTE INTERESES	DEUDA PENDIENTE
1	64666	45000	935334
2	67576	42090	867758
3	70617	39049	797141
4	73795	35871	723346
5	77115	32551	646231
6	80586	29080	565645
7	84212	25454	481433
8	88002	21664	393431
9	91962	17704	301469
10	96100	13566	205369
11	100424	9242	104945
12	104945	4723	0

COMO EFECTUAR LOS CALCULOS

Vamos a explicar de una forma general los aspectos de la aritmética financiera que hemos utilizado en esta parte del programa.

1. Amortización en cuota fijas

Al final del primer plazo deberemos todo el importe del capital prestado o deuda contraída más los intereses devengados durante ese período. Con el pago de la cuota abonaremos todos los intereses y con el sobrante amortizaremos algo del principal.

Al final del segundo período la situación será análoga, pero como la deuda será menor, también lo serán los intereses devengados, y al ser constante la cuota, la reducción de la deuda será mayor que en el caso anterior.

Al final del último período nos quedará una deuda tal que sumada a los intereses sea idéntica a la cuota establecida, con lo cual se habrá finalizado el proceso de amortización.

La fórmula general para el cálculo de capital más intereses es C*(1 + Ip/100) donde C es el capital solicitado o debido e Ip es el interés correspondiente al período elegido. Si el dinero lo prestan al 18% anual y los pagos tienen una periodicidad mensual Ip será 18/12 = 1.5 %.

Por simplicidad de expresión denominaremos A=(1+Ip/100) K=Cuota y N=Nº de plazos.

Fin Plazo No	Deuda+Int
1	C*A
2	(C*A-K)*A
.	....
.	....
N	....

Pago	Deuda Pendiente
K	C*A-K
K	C*A^2-K*A-K
K	...
K	...
K	(*)

(*) C*A^N-K*A^(N-1)-K*A^(N-2) -...-K*A-K

Considerando que la Deuda Pendiente al final del último pago debe ser cero, y ayudándonos del Cálculo Combinatorio obtendremos que:

```
    C*Ip*(1+Ip/100)^N
K=---------------------
     (1+Ip/100)^N-1
```

2. Capitalización con cuota fija

Al final de cada período dispondremos del capital que teníamos acumulado, más el importe de la cuota, más

los intereses devengados por el total durante el período entre plazos.

Operando en forma semejante al caso anterior llegaríamos a la siguiente fórmula que nos permite hallar la cuota necesaria para alcanzar un capital determinado en unas condiciones dadas:

```
          C*Ip/100
K=---------------------------
  (1+Ip/100)*((1+Ip/100)^N-1)
```

Para terminar os ofrecemos el listado del programa completo.

```
10 CLS:KEY OFF:COLOR 15,4,4
   :SCREEN 0:WIDTH  40
20 GOSUB 1540
30 'PRESTAMOS Y
   CAPITALIZACIONES
40 'MOLISOFT / INPUT MSX
50  'VARIABLES
60 'C=CAPITAL A FORMAR /
   PEDIR
70 'CA=CAPITAL ACUMULADO
80 'K=CUOTA PERIODICA
90 'P=No.DE MESES DE CADA
   PLAZO
100 'NP=No. DE PLAZOS
110 'I=INTERES ANUAL EN %
120 'DP=DEUDA PENDIENTE
130 'IK=PARTE INTERESES EN LA
    CUOTA
140 'CK=PARTE AMORTIZ. EN LA
    CUOTA
150 'MENU
160 CLS:PRINT TAB(5)
    "PRESTAMOS Y
    CAPITALIZACIONES":PRINT
    TAB(5)"==================
    ==========":PRINT
170 PRINT"1 - TOMA DE DATOS
    (CUOTA FIJA)":PRINT
180 PRINT"2 - VISUALIZACION
    TABLA DE PRESTAMOS":
    PRINT
190 PRINT"3 - IMPRESION TABLA
    DE   PRESTAMOS":PRINT
200 PRINT"4 - VISUALIZ. TABLA
    DE CAPITALIZACION":PRINT
210 PRINT"5 - IMPRESION TABLA
    DE CAPITALIZACION":PRINT
220 PRINT"6 - TOMA DE DATOS
    (CUOTA VARIABLE)":PRINT
230 PRINT"7 - VISUALIZ. TABLA
    DE CAPITALIZACION":PRINT
240 PRINT"8 - IMPRESION TABLA
    DE CAPITALIZACION":PRINT
250 LOCATE,22:PRINT"PULSE
    OPCION DESEADA"
260 P$=INKEY$:IF P$<"0" OR
    P$>"8" THEN 260
270 ON VAL(P$) GOTO 280,500,
    640,850,970,1100,1320,
    1430
280 REM  TOMA DE DATOS (CUOTA
    FIJA )
290 CLS
300 LOCATE,0:PRINT TAB(4)"===
    DATOS PROPORCIONADOS==="
310 LOCATE ,3: PRINT"CAPITAL
    (PTS.)    : "TAB(20) SPACE
    $(19)
320 LOCATE ,5: PRINT"INTERES
    ANUAL (%)  : "TAB(20)
    SPACE$(19)
330 LOCATE ,7: PRINT"NUMERO
    DE PLAZOS   : "TAB(20)
    SPACE$(19)
340 LOCATE ,9: PRINT"MESES
    POR PLAZO    : "TAB(20)
    SPACE$(19)
350 GOSUB 1620 :INPUT"CAPITAL
    ";C:LOCATE 20,3:PRINT
    SPACE$(19):LOCATE 20,3:
    PRINT INT(C+.5)
360 GOSUB 1620 :INPUT"INTERES
    ";I:LOCATE 20,5:PRINT
    SPACE$(19):LOCATE 20,5:
    PRINT(INT(I*100+.5))/100
370 GOSUB 1620 :INPUT"No. DE
    PLAZOS";NP:LOCATE 20,7:
    PRINT SPACE$(19):LOCATE
    20,7:PRINT INT(NP+.5)
380 GOSUB 1620 :INPUT"MESES
    POR PLAZO";P:LOCATE 20,9:
    PRINT SPACE$(19):LOCATE
    20,9:PRINT INT(P+.5)
390 IP=I*P/1200:CF=(1+IP)^NP
400 GOSUB 1620:P$="":INPUT
    "DESEA MODIFICAR S/N ";P$
    :IF P$="S" OR P$="s"
    THEN 350
410 GOSUB 1620:P$="":INPUT
    "PRESTAMO / CAPITALIZ. ?
    (P/C) ";P$:IF P$="P" OR
```

```
    P$="p" THEN 420 ELSE 780
420 REM CALCULO DE PRESTAMOS
430 LOCATE 5 ,13:PRINT"===
    DATOS CALCULADOS ==="
440 LOCATE 10,15:PRINT
    "( PRESTAMOS )"
450 K=C*IP*CF/(CF-1):K=INT
    (K+.5)
460 LOCATE ,17:PRINT "CUOTA
    PERIODICA (PTS)"
    TAB(22) K
470 LOCATE ,19:PRINT "TOTAL
    INTERESES (PTS)" TAB(22)
    INT(NP*K-C+.5)
480 GOSUB 1620:PRINT"PARA
    MENU PULSE CUALQUIER
    TECLA"
490 P$=INKEY$:IF P$="" THEN
    490 ELSE 150
500 REM " VISUALIZACION TABLA
    DE PRESTAMOS
510 DP=C
520 FOR B=1 TO NP STEP 15
530 CLS:PRINT TAB(8) "TABLA
    DE PRESTAMOS":PRINT
    TAB(8) "-----------------
    -":PRINT
540 PRINT"PLAZO"TAB(7)
    "PART.CAP."TAB(17)
    "PART.INT."TAB(28)"DEUDA.
    PEND.":PRINT STRING$
    (39,"="):PRINT
550 JB=(B+14):IF JB>NP THEN
    JB=NP
560 FOR J=B TO JB
570 IK=DP*IP:IK=INT(IK+.5)
    :CK=K-IK:CK=INT(CK+.5)
580 IF J=NP THEN CK=DP
590 DP=DP-CK :DP=INT(DP+.5)
600 PRINT J TAB(7) CK TAB(17)
    IK TAB(27) DP
610 NEXT J :GOSUB 1620:PRINT
    "CUALQUIER TECLA PARA
    SEGUIR"
620 IF INKEY$=""THEN 620
    ELSE NEXT B
630 GOTO 150
640 REM "IMPRESION TABLA DE
    PRESTAMOS
650 DP=C
660 FOR B=1 TO NP STEP 50
670 LPRINT:LPRINT:LPRINT:
    LPRINT TAB(28)"TABLA DE
    PRESTAMOS":LPRINT TAB(28)
    "-----------------":
    LPRINT
680 LPRINT TAB(5)"PRESTAMO:";
    C TAB(25) "PTS. AL:";I;"
    % DE INTERES":LPRINT
690 LPRINT TAB(5)"EN PAGOS DE
    ";K; " PTS. CADA ";P;
    " MES(ES) DURANTE ";NP;"
    PLAZOS":LPRINT
700 LPRINT TAB(8)"PLAZO"TAB
    (16)"PARTE CAPITAL"TAB
    (32)"PARTE INTERESES"TAB
    (40)"DEUDA PENDIENTE":
    LPRINT TAB(8) STRING$(57,
    "="):LPRINT
710 JB=(B+49):IF JB>NP THEN
    JB=NP
720 FOR J=B TO JB
730 IK=DP*IP:IK=INT(IK+.5):
    CK=K-IK:CK=INT(CK+.5)
740 IF J=NP THEN CK=DP
750 DP=DP-CK
760 LPRINT TAB(8)J  TAB(18)
    CK TAB(34) IK TAB(54) DP
770 NEXT J:LPRINT CHR$(12):
    NEXT B :GOTO 150
780 REM CALCULO DE
    CAPITALIZACIONES
790 LOCATE 5 ,13:PRINT"===
    DATOS CALCULADOS ==="
800 LOCATE 8,15:PRINT
    "( CAPITALIZACION )
810 K=C*IP/((1+IP)*(CF-1)):K=
    INT(K+.5)
820 LOCATE ,17:PRINT "CUOTA
    PERIODICA (PTS)" TAB(22)
    K
830 GOSUB 1620:PRINT"PARA
    MENU PULSE CUALQUIER
    TECLA"
840 P$=INKEY$:IF P$="" THEN
    840 ELSE 150
850 REM  VISUALIZACION TABLA
    DE CAPITALIZACIONES"
860 CA=0
870 FOR B=1 TO NP STEP 15
880 CLS:PRINT TAB(6)"TABLA DE
    CAPITALIZACIONES":PRINT
    TAB(6)"-----------------
    -------":PRINT
890 PRINT "PAGO No."TAB(10)
```

GANADORES DE LOS MEJORES DE INPUT MSX

En el sorteo correspondiente al número 7 entre quienes escribisteis mandando vuestros votos a LOS MEJORES DE INPUT han resultado ganadores:

NOMBRE	LOCALIDAD	JUEGO ELEGIDO
Victoria Planells Alargada	La Punta (Valencia)	Road Fighter
Sergio Menéndez Font	Esparraguera (Barcelona)	Alien 8
Gabriel Cabot Carbonell	Son Sardina (Baleares)	Jet Fighter
Iago González Moro	La Coruña	Knight Lore
Juan José Ortiz Mejías	Villarreal (Castellón)	Sir Fred
Rafael Martín Mateos	Málaga	Alien 8
C. Manuel Gasols Rodríguez	H. de Llobregat (Barcelona)	Knight Lore
J. Luis Benítez López	Ponferrada (León)	Green Beret
J. Manuel Romaris Martínez	S. de Compostela (Coruña)	Profanation
Marc Sabater Casas	Sabadell (Barcelona)	Knight Lore

```
    "CAPITAL ACUMULADO (FIN
    PLAZO)":PRINT STRING$
    (39,"="):PRINT
900 JB=(B+14):IF JB>NP THEN
    JB=NP
910 FOR J=B TO JB
920 CA=(CA+K)*(1+IP):CA=INT
    (CA+.5)
930 PRINT J  TAB(18)USING
    "##########"; CA
940 NEXT J :GOSUB 1620:PRINT
    "CUALQUIER TECLA PARA
    SEGUIR"
950 IF INKEY$=""THEN 950
    ELSE NEXT B
960 GOTO 150
970 REM " IMPRESION TABLA DE
    CAPITALIZACIONES"
980 CA=0
990 FOR B=1 TO NP STEP 50
1000 LPRINT:LPRINT:LPRINT:
     LPRINT TAB(25)"TABLA DE
     CAPITALIZACIONES":LPRINT
     TAB(25)"----------------
     ---------":LPRINT
1010 LPRINT TAB(5)"CAPITAL:";
     C TAB(25) "PTS. AL:";I;"
     % DE INTERES":LPRINT
1020 LPRINT TAB(5)"EN PLAZOS
     DE ";K; " PTS. CADA ";P;
     " MES(ES) DURANTE ";NP;"
     PLAZOS":LPRINT
1030 LPRINT TAB(13)"PAGO No."
     TAB(26)"CAPITAL
     ACUMULADO (FINAL PLAZO)"
     :LPRINT TAB(8)STRING$
     (57,"="):LPRINT
1040 JB=(B+49):IF JB>NP THEN
     JB=NP
1050 FOR J=B TO JB
1060 CA=(CA+K)*(1+IP):CA=INT(
     CA+.5)
1070 LPRINT TAB(15)J  TAB(33)
     USING"##########"; CA
1080 NEXT J:LPRINT CHR$(12):
     NEXT B
1090 GOTO 150
1100 REM  TOMA DE DATOS
     (CUOTA FIJA )
1110 CLS
1120 LOCATE,0:PRINT TAB(4)"==
     = DATOS PROPORCIONADOS =
     =="
1130 LOCATE ,3: PRINT"CUOTA
     INIC.MENSUAL: "TAB(20)
     SPACE$(19)
1140 LOCATE ,5: PRINT
     "INCREMENTO ANUAL %: "
     TAB(20) SPACE$(19)
1150 LOCATE ,7: PRINT"NUMERO
     DE PLAZOS   : "TAB(20)
     SPACE$(19)
1160 LOCATE ,9: PRINT"INTERES
     CAPITAL(%):"TAB(20)
     SPACE$(19)
1170 GOSUB 1620 :INPUT"CUOTA
     INICIAL ";K:LOCATE 20,3:
     PRINT SPACE$(19):LOCATE
     20,3:PRINT INT(K+.5)
1180 GOSUB 1620 :INPUT"%
     INCREMENTO ANUAL ";IC:
     LOCATE 20,5:PRINT SPACE$
     (19):LOCATE 20,5:PRINT
     (INT(IC*100+.5))/100
1190 GOSUB 1620 :INPUT"No. DE
```

```
        PLAZOS ";NP:LOCATE 20,7:
        PRINT SPACE$(19):LOCATE
        20,7:PRINT INT(NP+.5)
1200 GOSUB 1620:INPUT"INTERES
        ANUAL CAP. (%) ";I:
        LOCATE 20,9:PRINT SPACE$
        (19):LOCATE 20,9:PRINT
        (INT(I*100+.5))/100
1210 IP=I/1200
1220 GOSUB 1620:P$="":INPUT
        "DESEA MODIFICAR S/N ";
        P$:IF P$="S" OR P$="s"
        THEN 1170
1230 REM  CALCULO DE
        CAPITALIZACIONES
1240 LOCATE 5 ,14:PRINT"===
        DATOS CALCULADOS ==="
1250 LOCATE 7,16:PRINT
        "( CAPITALIZACIONES )"
1260 CA=0:NK=K
1270 FOR L=1 TO NP STEP 12:IF
        (NP-L)=>12 THEN HB=12
        ELSE  HB=(NP-L)+1
1280 FOR H=1 TO HB:CA=(CA+NK)
        *(1+IP):NEXT H:NK=NK*
        (1+IC/100):NEXT L
1290 LOCATE ,19:PRINT"CAPITAL
        ALCANZADO (PTS)" TAB(22)
        INT(CA+.5)
1300 GOSUB 1620:PRINT"PARA
        MENU PULSE CUALQUIER
        TECLA"
1310 P$=INKEY$:IF P$="" THEN
        1310 ELSE 150
1320 REM  VISUALIZACION TABLA
        DE CAPITALIZACIONES"
1330 CA=0 :NK=K
1340 FOR B=1 TO NP
1350 IF B MOD 15<>1 THEN 1370
        ELSE CLS:PRINT " TABLA
        DE CAPITALIZACIONES
        (CUOTA VAR.)":PRINT "---
        ------------------------
        ------------":PRINT
1360 PRINT "PAGO No."TAB(10)
        "CUOTA" TAB(20)"CAPITAL
        ACUMULADO":PRINT STRING$
        (39,"="):PRINT
1370 IF B>1 AND B MOD 12 =1
        THEN NK=NK*(1+IC/100):NK
        =INT(NK+.5)
1380 CA=(CA+NK)*(1+IP):PRINT
        B TAB(10)NK TAB(18)
        USING"##########";
        INT(CA+.5)
1390 IF B MOD 15=0 THEN
        GOSUB 1410
1400 NEXT B:GOSUB 1410:
        GOTO 150
1410 GOSUB 1620:PRINT
        "CUALQUIER TECLA PARA
        SEGUIR"
1420 IF INKEY$="" THEN 1420
        ELSE RETURN
1430 REM  IMPRESION TABLA DE
        CAPITALIZACIONES (CUOTA
        VARIABLE)
1440 CA=0 :NK=K
1450 FOR B=1 TO NP
1460 IF B MOD 50<>1 THEN 1500
        ELSE CLS:LPRINT:LPRINT:
        LPRINT:LPRINT TAB(15)
        "TABLA DE CAPITALIZACION
        ES (CUOTA VARIABLE)":
        LPRINT TAB(15)"---------
        ------------------------
        ---------":LPRINT
1470 LPRINT TAB(5)"CUOTA
        INICIAL DE :";K;"PTS CON
        INCREMENTO ANUAL DEL ";
        IC;" % "
1480 LPRINT TAB(5)"PAGADERAS
        CADA MES DURANTE ";NP;
        " MESES":LPRINT
1490 LPRINT TAB(12)"PAGO No."
        TAB(28) "CUOTA" TAB(45)
        "CAPITAL ACUMULADO":
        LPRINT TAB(12) STRING$
        (50,"="):LPRINT
1500 IF B>1 AND B MOD 12 =1
        THEN NK=NK*(1+IC/100):NK
        =INT(NK+.5)
1510 CA=(CA+NK)*(1+IP):LPRINT
        TAB(16)B TAB(27) USING
        "######";NK;:LPRINT TAB
        (47) USING"##########";
        INT(CA+.5)
1520 IF B MOD 50=0 THEN
        LPRINT CHR$(12)
1530 NEXT B:LPRINT CHR$(12):
        GOTO 150
1540 CLS: LOCATE ,5:PRINT
        STRING$(39,"*")
1550 FOR I=1 TO 15:PRINT "*"
        TAB(38) "*":NEXT I
1560  LOCATE ,20:PRINT
        STRING$(39,"*")
1570 LOCATE 10,7:PRINT
        "P R E S T A M O S":
        LOCATE 18,9:PRINT"Y":
        LOCATE 4,11:PRINT
        "C A P I T A L I Z A C I
        O N E S"
1580 LOCATE 10,15:PRINT
        "I N P U T  M S X"
1590 LOCATE 20,18:PRINT
        "Molisoft 1986"
1600 FOR I=1 TO 5000:NEXT I:
        RETURN
1610 STOP
1620 LOCATE ,22:PRINT SPACE$
        (39):LOCATE,22:RETURN
1630 SAVE"PRESYCAP.BAS"
```




RELACIONES LOGICAS

- OPERADORES RELACIONALES
- ¿UNA COSA ES MAYOR, IGUAL O MENOR QUE OTRA?
- OPERADORES LOGICOS
- USO DE AND, OR Y NOT

Los ordenadores son capaces de ejecutar millones de operaciones lógicas por segundo. Pero la lógica es además una parte fundamental de cualquier programa.

Hay tres tipos de expresiones que se utilizan en la programación en lenguaje BASIC. Una gran parte de todo programa está constituída por operaciones aritméticas y las realizadas con cadenas de caracteres; sin embargo, es el tercer tipo de operaciones, las realizadas con las expresiones lógicas, el que realmente se ocupa de la toma de decisiones dentro de un programa.

Su función elemental consiste en evaluar si algo es «verdadero» o «falso». Las palabras y símbolos que utiliza el programa para hacer esto se llaman operadores (en algunos casos se llaman también conectores, o conectivas).

En las expresiones lógicas se utilizan dos tipos de operadores: los relacionales (que incluyen símbolos matemáticos tales como <, > e =), y los lógicos que forman parte del repertorio de palabras reservadas en todas las versiones del BASIC: AND, OR y NOT. El BASIC **MSX** dispone además de las palabras XOR, IMP y EQV.

MAYOR, IGUAL O MENOR

Los operadores relacionales pueden utilizarse incluso en los programas BASIC más sencillos. Las posibles comparaciones que se pueden llevar a cabo con estos operadores son:

A > B ... A es mayor que B
A < B ... A es menor que B
A ⩾ B ... A es mayor o igual que B
A ⩽ B ... A es menor o igual que B
A = B ... A es igual que B
A <> B ... A es diferente de B

Habrás visto estos operadores utilizados docenas de veces en los programas de **INPUT** o en cualquier otra parte, y aunque su utilización es múltiple, su valor en las pruebas condicionales resulta más evidente en relación con la sentencia IF ... THEN. Un ejemplo típico de esto es:

IF A > B THEN PRINT «A ES MAYOR QUE B»

En este caso, el valor de la variable A debe superar el valor de B para que se ejecute el resto de la línea. En los casos en que el valor de A permanezca por debajo del valor de B, el programa se limitará a pasar a la línea siguiente. Con esto es posible disponer de una cierta capacidad de salto condicional: si resulta un determinado conjunto de valores, el programa hace una cosa; si resulta otro conjunto de valores, el programa hace otra cosa distinta.

Los operadores relacionales no están únicamente limitados a manejar valores numéricos; también sirven para comparar cadenas de caracteres. No obstante, hay que manejarlos con precaución para evitar encontrarse con resultados inesperados. Se debe tener presente que la comparación siempre se detiene en el número de caracteres correspondiente a la más corta de las dos cadenas que se comparan. Así, si una de las cadenas contiene once caracteres, mientras que la otra sólo contiene siete, únicamente intervendrán en la comparación los primeros siete caracteres de la cadena más larga, comparándose con sus correspondientes en la cadena corta.

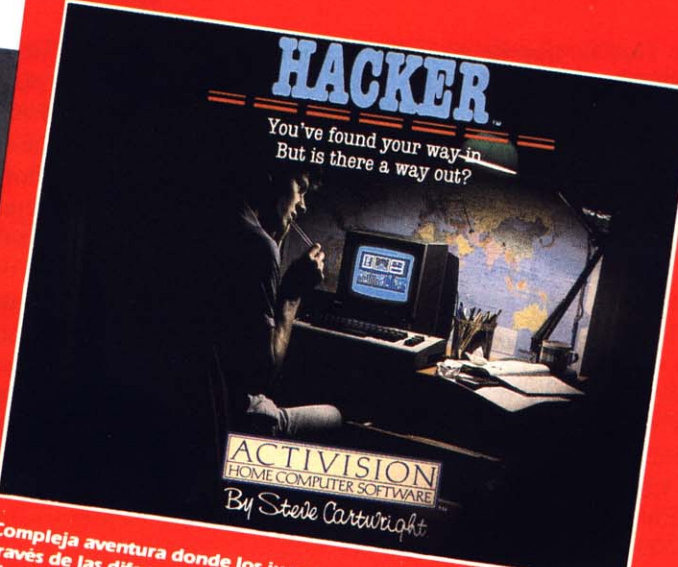
Los caracteres se comparan uno por uno y de izquierda a derecha; de aquí pueden surgir los resultados inesperados. En una comparación de tipo numérico, la proposición 5 > 10 es evidentemente falsa, pero en una comparación entre cadenas, puedes encontrarte una sentencia en la que se comparen dos variables en la siguiente forma: A$ > B$. Si A$ = «5» y B$ = «10» el ordenador compara en primer lugar el carácter que figura más a la izquierda, que en este caso es un 5 en una de las cadenas y un 1 en la otra, y a continuación se para porque considera que ya no tiene nada que comparar.

En consecuencia se obtiene un «verdadero» como resultado de la comparación, debido a que 5 es mayor que 1, pero el ordenador ignora los números restantes de la cadena más larga. Tienes que estar atento para no cometer errores de este tipo.

En las comparaciones entre cadenas de caracteres, a las letras del alfabeto se les supone la siguiente relación de orden: «A» < «B» < «C» < «D» , etcétera. Pero también aquí has de tener cuidado, ya que el ordenador no entiende realmente lo que estos caracteres significan. Lo que hace es comparar los códigos correspondientes a cada carácter, siendo igualmente significativos dentro de la cadena los espacios y otros caracteres que no son letras, pero que disponen también de un código propio. Como puedes imaginarte, esto podría causar estragos en un programa de clasificación de cadenas por orden alfabético.

Cuando ambas cadenas contienen una secuencia de caracteres parcialmente idéntica, se considera numéricamente mayor a la cadena más larga. Pero observa que una cadena más corta es considerada mayor en una expresión como la siguiente: «ABD» > «ABCD» ya que la comparación se detiene en cuanto se encuentra con la primera diferencia, es decir, al comparar los códigos correspondientes a los caracteres «D» y «C». El funcionamiento es de hecho el mismo que se sigue al asignar prioridades alfabéticas

TE PRESENTAMOS LOS IMPRESCINDIBLES PARA TU ORDENADOR



Compleja aventura donde los jugadores deben buscar a través de las diferentes pistas y problemas como resolver el misterio.

CSAM



El más espectacular KARATE hasta ahora nunca visto. Practica las artes marciales en distintos escenarios del Mundo.
"ZAPP" dijo que era el mayor desafío en juegos de este tipo.

CM



El más reciente avance técnico. Vd. puede realizar una jornada completa dentro de una cápsula espacial desde que se levanta de la superficie de la tierra y acude a un encuentro en el espacio, hasta que aterriza nuevamente. Comprueba tu habilidad.

CSM

Disponibles para:

COMMODORE	C
SPECTRUM	S
AMSTRAD	A
MSX	M

EN TIENDAS ESPECIALIZADAS Y GRANDES ALMACENES, O DIRECTAMENTE POR CORREO O TELEFONO A: **PROEIN, S.A.**

Distribuido en Cataluña por: DISCOVERY INFORMATIC C/. Arco Iris, 75 - BARCELONA - Tels. 256 49 08 / 09

Velázquez, 10 - 28001 Madrid - Tels. (91) 276 22 08/09

en la redacción de un índice o un diccionario, en el que «para» figura antes que «paradoja», pero después de «pan».

VERDADERO O FALSO

Toda comparación produce un resultado que en realidad es un número entero. Cuando la comparación resulta ser falsa, el resultado es 0, mientras que cuando es verdadera el resultado es −1. Teclea este comando en modo directo: PRINT 6>5, 5>7

La expresión de la izquierda es verdadera y la de la derecha falsa. En consecuencia obtendrás en la pantalla un −1 y un 0.

Los números enteros que obtengas como resultado de las comparaciones, puedes utilizarlos para realizar cálculos dentro de tus programas. No obstante tienes que tener cuidado de no intentar hacer divisiones por 0, lo que te daría un mensaje de error.

OPERADORES LOGICOS

Las **palabras reservadas** AND, OR y NOT son operadores lógicos que constituyen una poderosa extensión a la capacidad de toma de decisiones por medio de la sentencia IF ... THEN. Por ejemplo puedes poner: IF esto es cierto AND eso otro es cierto, THEN hacer una determinada cosa. De esta forma puedes ir construyendo funciones de gran complejidad. Estos operadores, que a veces se llaman también operadores **booleanos** (aunque no debes dejarte impresionar por este nombre) se pueden utilizar en comandos directos o dentro de los programas para obtener un «valor de verdad» en condiciones de prueba que pueden llegar a ser muy complejas. De esta forma constituyen una alternativa muy cómoda a lo que en otro caso sería un confuso conglomerado de saltos condicionales a base de una utilización masiva de sentencias IF ... THEN.

USO DE «AND»

En su forma más simple, se puede considerar que el operador AND tiene el mismo significado que la conjunción española «y». En una expresión tal como la siguiente:

```
IF V > 0 AND V < 100 THEN PRINT «DENTRO DE MARGENES»
```

El mensaje sólo aparecerá en pantalla cuando la variable V esté comprendida entre 0 y 100.

En un programa puedes tener una línea como ésta:

```
990 OK=-1 AND MES>5 AND
    MES>10 AND ANO>1900
    AND ANO<2001
```

Esta línea de programa te permitiría comprobar si una determinada fecha corresponde a un verano del siglo XX.

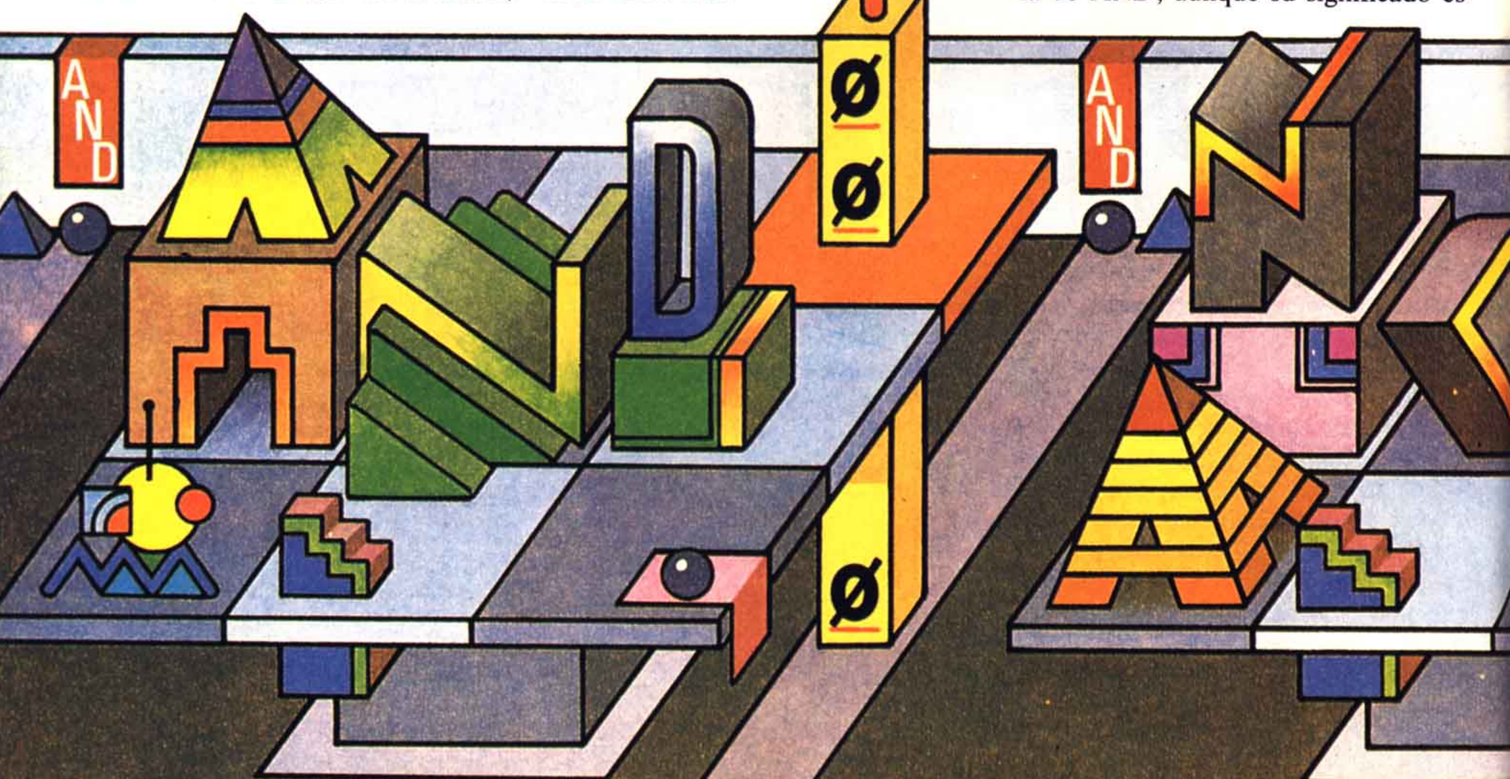
En este caso se utiliza AND para realizar cuatro pruebas, cuyos resultados deben ser todos ciertos para que OK siga siendo verdadera. En esta prueba de validación se asigna en primer lugar, a la variable OK, el valor «verdadero». A continuación se impone la condición de que MES caiga en el intervalo de 6 a 9, y de que AÑO esté entre 1901 y 2000. Pero miremos un poco más en detalle lo que ocurre. Si se cumplen todas las condiciones, todas las expresiones darán un valor verdadero. Así, de hecho la línea de programa se convierte en :

```
990 OK=-1 AND -1 AND -1
    AND-1 AND -1
```

Si ahora le añades PRINT OK, puedes comprobar que el resultado es en efecto −1, es decir verdadero.

USO DE «OR»

El operador OR puede recibir una consideración en gran parte análoga a la de AND, aunque su significado es



diferente, como ocurre con la conjunción española «o» en su uso cotidiano. También aquí observaremos su empleo en una sencilla línea de programa en la que se utiliza para comparar los valores de una variable:

```
990 IF V=8 OR V=10 THEN
    PRINT "OK"
```

El mensaje aparecerá en la pantalla si el valor de la variable V es 8 o 10.

El operador OR resulta de gran utilidad cuando se comprueba la validez de las entradas suministradas a una sentencia INPUT dentro de un programa. Si por ejemplo el programa te pide que introduzcas tu edad, siempre habrá alguien que tecleará —10 ó 999 con la intención de probar a ver qué pasa. Pero puedes solucionar el problema de la siguiente forma:

```
10 INPUT A
20 IF A<1 OR A>120 THEN
   PRINT"POR FAVOR NO TE
   PASES":GOTO 10
```

USO DE «NOT»

El tercer operador lógico que se utiliza con frecuencia es el NOT. Es un poco diferente de los anteriores en el sentido de que sólo utiliza un argumento, es decir, sólo actúa sobre la expresión numérica o lógica que le sigue, a diferencia de AND y OR que actúan sobre la expresión que les precede y sobre la que les sigue, es decir son operadores binarios o de dos argumentos.

Puedes utilizar NOT en un programa como el siguiente:

```
990 IF NOT(A>10) THEN 999
```

En el lenguaje hablado normal, podríamos traducir la anterior expresión más o menos así: «Si el valor de A no es mayor que 10, pasa a la línea 999».

La función lógica que realiza el operador NOT es convertir una condición verdadera en falsa y viceversa. Puedes utilizarla pues para invertir el valor resultante de una prueba anterior.

TABLAS DE VERDAD

En relación con los operadores lógicos, te encontrarás con mucha frecuencia con las «tablas de verdad».

En realidad son algo muy sencillo que se utiliza para dar una representación visual de lo que ocurre cuando se hacen operaciones de AND y OR con los valores −1 y 0 (verdadero y falso). El operador NOT no necesita en realidad una tabla de verdad, ya que basta con invertir los valores para observar su efecto.

Las tablas de verdad pueden adoptar formas diferentes. Una forma típica para el operador AND es la siguiente:

```
 A  B  C
------------------
-1 -1 -1   (LINEA 1)
-1  0  0   (LINEA 2)
 0 -1  0   (LINEA 3)
 0  0  0   (LINEA 4)
```

El significado de la línea 1 de la tabla es el siguiente: «Si A es verdadera y B es verdadera, entonces C es verdadera». El significado de la línea 2 es: «Si A es verdadera y B es falsa, entonces C es falsa». La línea 3 significa: «Si A es falsa y B es verdadera, entonces C es falsa». La línea 4 por su parte significa: «Si A y B son falsas, entonces C es falsa». Análogamente la tabla de verdad del operador OR es:

```
 A  B  C
---------
-1 -1 -1
-1  0 -1
 0 -1 -1
 0  0  0
```

La primera línea significa: «Si A es verdadera y B es verdadera, entonces C es verdadera». Las líneas 2 y 3 pueden ser ambas interpretadas como: «Si A o B son verdaderas, entonces C es



verdadera». El significado de la línea 4 es: «Si A y B son ambas falsas, entonces C es también falsa».

APLICACION PRACTICA

Aquí tienes finalmente un programa que utiliza los operadores AND, OR y NOT. Se trata de una versión simplificada de un programa que calcula la escala correcta de salarios para alguien que ha solicitado un empleo:

```
10 INPUT"EDAD";EDAD
20 INPUT"ANOS EN LA EMPRESA";
   ONIVEL
30 MAS18=(EDAD>=18)
40 CUAL =(ONIVEL>=5)
50 IF NOT MAS18 AND NOT CUAL
   THEN PRINT"NO VALIDO"
60 IF (NOT MAS 18 AND CUAL)
   OR (MAS18 AND NOT CUAL)
   THEN PRINT "ESCALA DE
   SALARIO UNO"
70 IF MAS18 AND CUAL THEN
   PRINT"ESCALA DE SALARIO
```

Supongamos que como respuesta a las dos primeras sentencias INPUT, tecleas 20 años de edad y cuatro en la empresa. Esto significa que (EDAD ⩾ 18) es verdad, mientras que (ONIVEL ⩾ 5) es falso. En consecuencia la persona es MAS18 (más de 18 años) y NO CUAL (no cualificada). El ordenador encuentra enseguida este conjunto de condiciones en la línea 60, imprimiendo a continuación «ESCALA DE SALARIO UNO». Puedes ampliar fácilmente este programa para que compruebe más condiciones, por ejemplo si el solicitante cuenta con más de dos años de experiencia, o cualquier otra circunstancia que sea necesaria para el trabajo.

OPERACIONES NUMERICAS

El uso de los operadores lógicos no está únicamente limitado a las sentencias IF ... THEN que acabamos de ver. También se pueden utilizar en algunos tipos de operaciones numéricas. Puede que ya las hayas visto utilizadas así en algún programa y a lo mejor te has quedado maravillado de lo que sucedía. De hecho no siempre funcionan de la forma obvia que podría esperarse. Teclea lo siguiente en modo inmediato:

PRINT 375 AND 47

Tal vez esperas que el resultado sea 422, o quizás 37547. Y sin embargo el resultado es 39. ¿Qué ha sucedido? Para explicarlo, tenemos que adentrarnos algo más en la forma de trabajar del ordenador, a fin de entender el modo en que éste maneja los dos argumentos cuyo AND se calcula.

En principio ambas expresiones (375 y 47) se convierten en su equivalente binario.

375 en binario es:
0000000101110111

Por su parte, 47 en binario es:
0000000000101111

A continuación la función AND realiza una comparación bit a bit sobre los 16 bits, dando un 1 como resultado únicamente cuando hay un 1 en el mismo bit de ambos números en binario. Empezando por la derecha, sólo los pares de bits cero, 1, 2 y 5 cumplen esta condición. Esto da como resultado el número binario 0000000000100111.

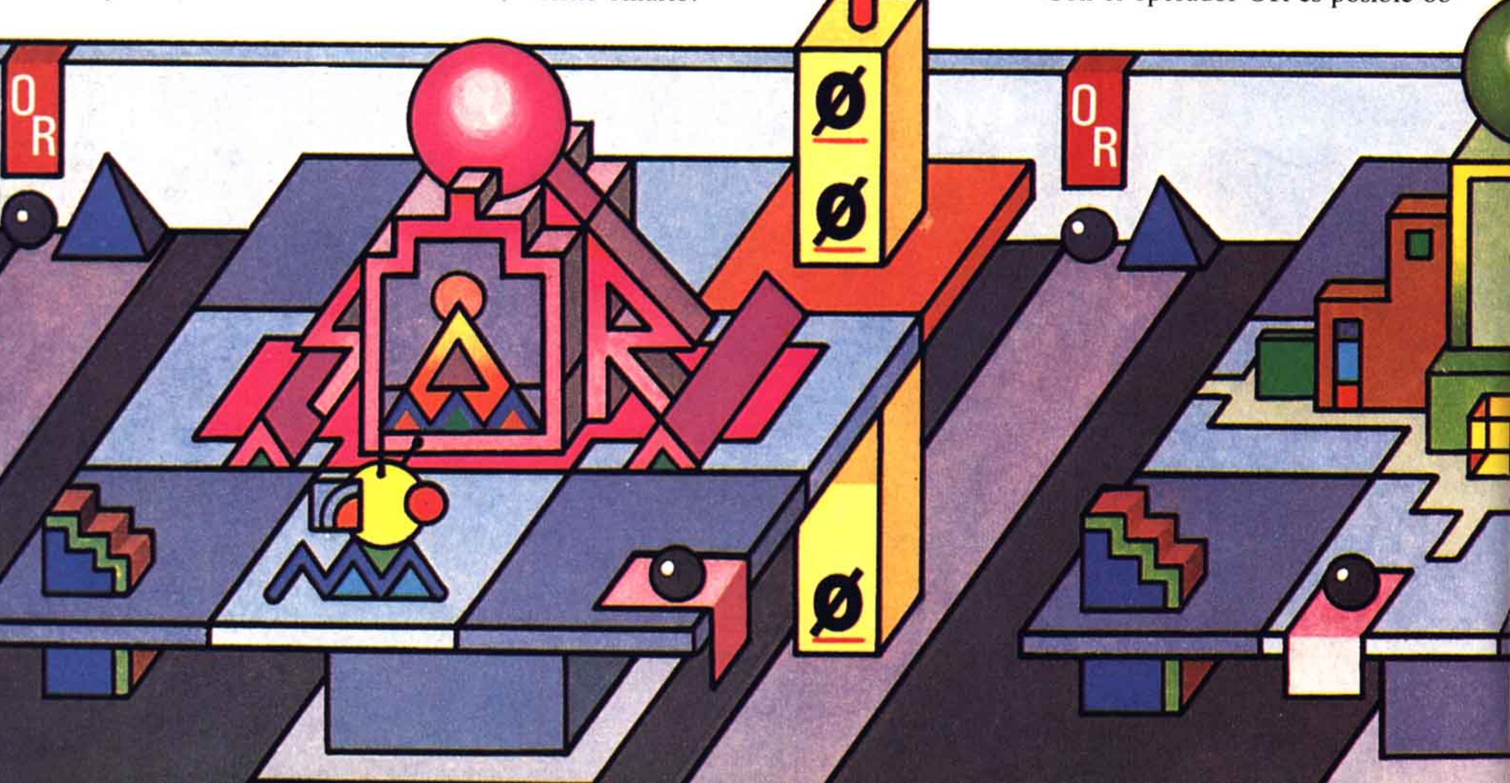
Al volver a hacer la conversión a decimal, los bits 5, 2, 1 y 0 puestos a 1 dan un valor combinado de 39; éste es el resultado de hacer un AND de 375 y 47.

Ensaya ahora el siguiente comando en modo directo:

PRINT 375 OR 47

¿A qué se debe el resultado 383 que se obtiene en este caso? Al realizar la operación con OR lógico el resultado de cada comparación bit a bit es 1 si hay un 1 en cualquiera de los bits del par que se compara. Observa de nuevo la forma binaria de los números anteriores y verás que los bits 8, 6, 5, 4, 3, 2, 1 y 0 tienen un 1 en al menos uno de los bits del par. El número binario resultante es 0000000101111111 que es 383 en decimal.

Con el operador OR es posible ob-



tener el mismo valor de bits con expresiones diferentes. Por ejemplo, si tecleas PRINT 315 OR 75, el resultado es también 383. Si quieres puedes comprobarlo realizando la conversión de los números a forma binaria.

El valor −1 (que se almacena como FFFFFFFF en hexadecimal, es decir un conjunto de bits que son todos 1) queda inalterado cuando con él se realiza una operación de OR lógico con otro número cualquiera. Para hacer la prueba, teclea el siguiente comando directo:

PRINT −1 OR 375

cuyo resultado es −1.

Veamos ahora una aplicación numérica para el operador NOT:

PRINT NOT 10.75,NOT −11

Los valores resultantes son −11 y 10. Así vemos que en efecto NOT suma 1 al número sobre el que se aplica y a continuación cambia su signo. Observa que el número en punto flotante se redondea por defecto al entero más próximo y que es posible estimar el valor real por medio de $X = -(X + 1)$. De hecho este valor se convierte a un entero de cuatro bytes, cuyos bits quedan invertidos.

USO DE XOR, IMP Y EQV

El BASIC **MSX** dispone, además de los operadores AND, OR y NOT, que se encuentran en casi todas las versiones de BASIC, de otros tres operadores: XOR, IMP y EQV, que comentamos a continuación.

XOR u OR exclusivo compara dos números binarios bit a bit. Si los dos bits son distintos, el resultado es verdadero, y si son iguales, falso. Uno de los usos más interesantes de XOR es el de convertir las letras mayúsculas en minúsculas y viceversa. Observa que el código ASCII de la letra A es 65, que en binario se escribe 0000000001000001 y que el código de la letra a (minúscula) es 97: en binario 0000000001100001. Todas las letras mayúsculas tienen el 5º bit (el 6º dígito empezando por la derecha) a 0, mientras que todas las letras minúsculas lo tienen a 1, y ésta es la única diferencia entre cada letra mayúscula y su homóloga minúscula. Por tanto basta tener en cuenta que

```
0000 0000 0100 0001 XOR
0000 0000 0010 0000 =
-----------------------
0000 0000 0110 0001

0000 0000 0100 0010 XOR
0000 0000 0010 0000 =
-----------------------
0000 0000 0110 0010
```

etc..., para entender el siguiente programa:

```
10 A$=INPUT$(1)
20 PRINT CHR$(ASC(A$)XOR 32)
30 GOTO 10
```

De esta forma introduciendo la letra A, el programa te devolverá la a, y, en general, introduciendo una letra mayúscula el programa te devuelve la misma letra, pero ahora minúscula, y viceversa. (En realidad puedes utilizar el operador OR para obtener el mismo resultado).

Ya habrás observado que la tabla de verdad del operador XOR es:

```
 A  B  C
---------
-1 -1  0
-1  0 -1
 0 -1 -1
 0  0  0
```

La propiedad exclusiva de XOR es que si se usa dos veces, se obtiene el valor original:

A XOR B XOR B = A

Los operadores IMP y EQV son menos utilizados que los anteriores y en realidad se pueden obtener como combinación de ellos (lo mismo ocurre con XOR).

La tabla de verdad de IMP es:

```
 A  B  C
---------
-1 -1 -1
-1  0  0
 0 -1 -1
 0  0 -1
```

y se puede poner en función de OR y NOT, como sigue:

A IMP B = (NOT A) OR B

El efecto de EQV es exactamente el opuesto de XOR. C es verdadero cuando A y B son verdaderos, y falso en caso contrario. La tabla de verdad de EQV es:

A	B	C
-1	-1	-1
-1	0	0
0	-1	0
0	0	-1

y se puede escribir en función de AND, OR y NOT de la siguiente forma:

A EQV B = (A AND B) OR ((NOT A) AND (NOT B))

OPERACIONES AVANZADAS CON BITS

Los operadores lógicos tienen una función adicional en los ordenadores **MSX** como operadores bit a bit capaces de controlar las funciones del ordenador. Algunas posiciones de memoria de los ordenadores **MSX** realizan funciones determinadas. Puedes especificar una determinada misión haciendo que cierta combinación particular de bits esté a uno, haciendo, a la vez, que otros bits se pongan a cero. Los comandos utilizados en BASIC para controlar el contenido de la memoria son POKE y PEEK. Supón que en la dirección 50001 tienes el número 55. Una instrucción típica con POKEs y PEEKs puede tener la forma:

POKE 50001, PEEK(50001) AND 251

Esto hace que el bit 2 se ponga a cero. PEEK(55001) dará el valor 55 (decimal), por tanto veamos que sucede al ejecutar la operación AND:

Valor 55 : 00110111

Valor 251: 11111011

Operando bit a bit se tiene el número 00110011 (=51). Compara la antigua configuración del valor 55 con la del valor 51 (nuevo valor PEEK de la dirección 55001) y podrás ver que el bit 2 (tercero por la derecha) a pasado a ser 0. Puedes volver a la situación inicial utilizando el operador OR:

POKE 55001,PEEK (1) OR 4

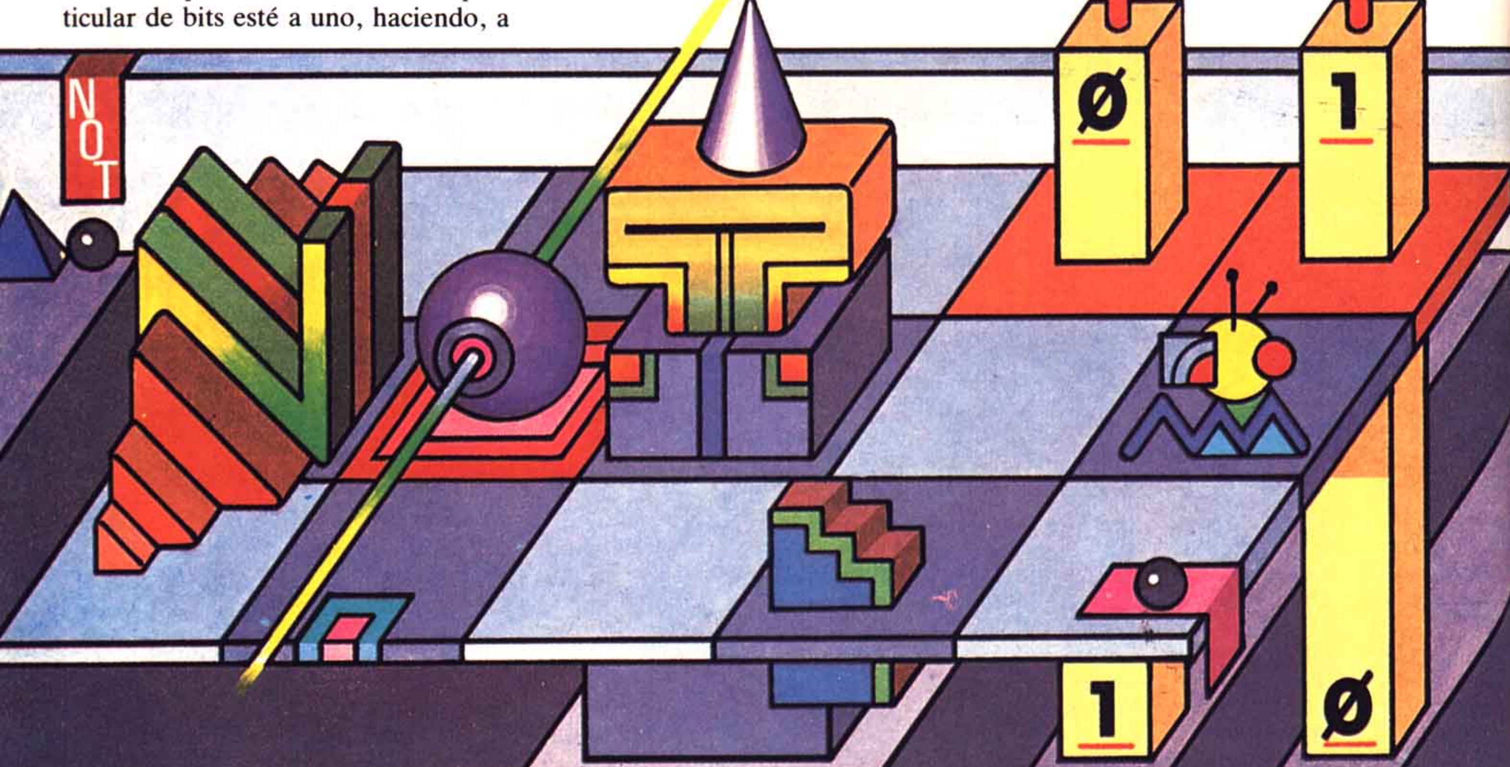
Al aplicarle al número binario 00110011 (51) el operador OR con el número binario 00000100 (4) obtenemos el número 00110111, que era el valor inicial del PEEK antes de realizar la primera operación —en otras palabras, el valor almacenado en la dirección 55001, vuelve a valer 55.

Puede parecerte que esto es en realidad un poco complicado, cuando puedes cambiar el valor de una dirección utilizando directamente POKE para introducir el nuevo valor. Sin embargo, corres el riesgo de modificar otros bits cambiando su estado original. Haciendo uso del OR lógico sólo tienes que especificar el bit que quieres poner a 1, dejando así inalterados los bits restantes.

De igual forma puedes utilizar AND para poner a 0 uno o más bits, simplemente sumando los valores decimales de los bits que quieres poner a cero y restando la suma de 255. Así, para poner a 0 los bits 0 y 2 por ejemplo, basta con sumar los números 1 y 4 (00000001 y 00000100 respectivamente) y restar la suma (5) de 255. Por tanto basta con que uses AND 250. Para volverlos a poner a 1 basta que hagas OR 5.

Este procedimiento se conoce como enmascaramiento de bits y la diferencia con el uso directo de POKE es que en este caso, se cambia el valor de la dirección afectada por el nuevo valor, mientras que los operadores lógicos no cambian necesariamente todos los bits del número sino sólo los que tu quieras cambiar.

Observa, por otra parte, que los operadores lógicos sólo trabajan con números enteros, es decir, con números dentro del rango −32768 a + 32767.



POR DELANTE Y SIEMPRE AVANZANDO **MITSUBISHI ELECTRIC**

Nuevo precio 56.696 + IVA.



ML-G1

la viva imagen del profesional.

La nueva generación MITSUBISHI MSX 2 sistema compacto, conjuga las características domésticas con las prestaciones más profesionales.
MITSUBISHI. El concepto informático en pleno avance.

LA MEMORIA DE VIDEO DE MSX: LOS MODOS DE TEXTO

Te vamos a contar todos los secretos de los modos de texto de tu MSX, SCREEN 0 y SCREEN 1, para que puedas hacer desde ellos todo tipo de diabluras, ya sea en BASIC o en lenguaje máquina.

En anteriores números de INPUT hemos visto cómo está distribuida la memoria de vídeo (VRAM) en los modos SCREEN 2 y SCREEN 3. Veremos en esta ocasión cómo se distribuye esta misma memoria en los modos SCREEN 0 y SCREEN 1. Estos modos son eminentemente de texto, por lo que, aunque se pueden dibujar gráficos con ellos, estos son de muy baja resolución. Sin embargo son los modos de pantalla más utilizados y con lo que resulta más sencillo establecer la comunicación con el ordenador.

Empezaremos viendo el modo 0, para establecer posteriormente las analogías y diferencias entre éste y el modo 1, así como la posibilidad de manipular los colores y de hacer uso de *sprites*, en este último modo de pantalla.

LA TABLA DE NOMBRES DE PATRONES EN SCREEN 0

Esta tabla, que tiene un nombre tan largo, ocupa sólamente 960 bytes, y se ocupa de contener los datos necesarios para la presentación del texto en la pantalla. En el modo 0 podemos suponer la pantalla dividida en 40 columnas y en 24 filas, de forma que tenemos en total 40 * 24 = 960 cuadros. Así, a cada cuadro le corresponde un byte de la tabla de nombres.

En cada uno de los bytes de esta tabla hay un número que representa el código ASCII del caracter que debe aparecer en una posición determinada de la pantalla.

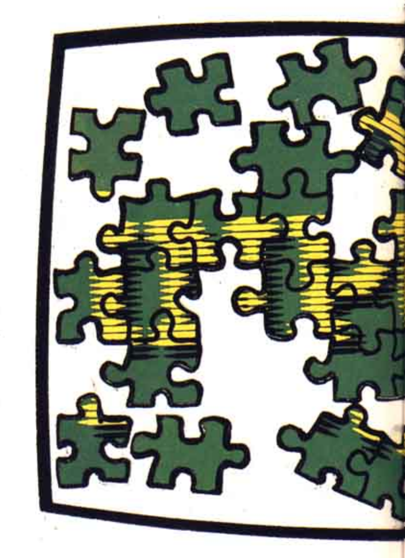
En la figura 1 puedes ver como se asignan esas posiciones de memoria a cada cuadro de la pantalla. Si numeramos las filas de 0 a 23 y las columnas de 0 a 39, al cuadro (0,0) le corresponde el primer byte de la tabla, que se obtiene mediante la variable BASE(0) que es igual a 0. Por tanto al cuadro (0,0) le corresponde el byte 0, y el número que contenga dicho byte es el código ASCII del caracter que ocupa el cuadro. Si por ejemplo en el byte 0 está el valor 65 (decimal), el carácter del cuadro (0,0) será la letra A mayúscula. En el cuadro (1,0) (a la derecha del anterior) estará el caracter correspondiente al byte 1, en el cuadro (2,0), el del byte 2, ... y así hasta el cuadro (39,0) que se corresponde con el byte 39. El siguiente cuadro es el cuadro (0,1), primer cuadro de la segunda fila, inmediatamente debajo del cuadro (0,0), y le corresponde el byte 40... Procediendo de la misma forma, llegaremos al cuadro (39,23) que es el de la esquina inferior derecha, al que le corresponde el byte 960 de la tabla. (figura 1).

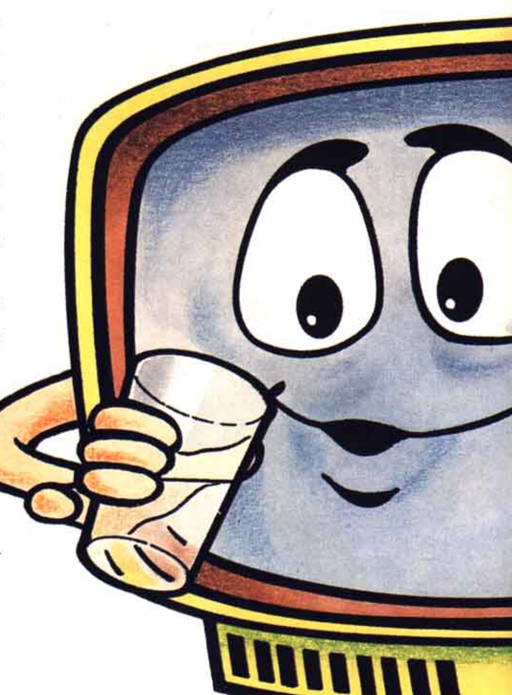
La instrucción WIDTH hace que se dejen márgenes a la izquierda y a la derecha, lo que significa que habrá ciertas columnas en las que no se escriba nada. Pero esto no hace que la tabla 0 sea más corta, sencillamente se ignoran los bytes correspondientes a esas columnas.

Veamos un ejemplo. Teclea el siguiente programa:

```
10 SCREEN 0
20 WIDTH 39
30 LOCATE 0,0
40 PRINT"HOLA"
50 FOR I%=0 TO 5
60 PRINT VPEEK(I%)
70 NEXT I%
```

Al correrlo aparecerá en la pantalla lo siguiente:



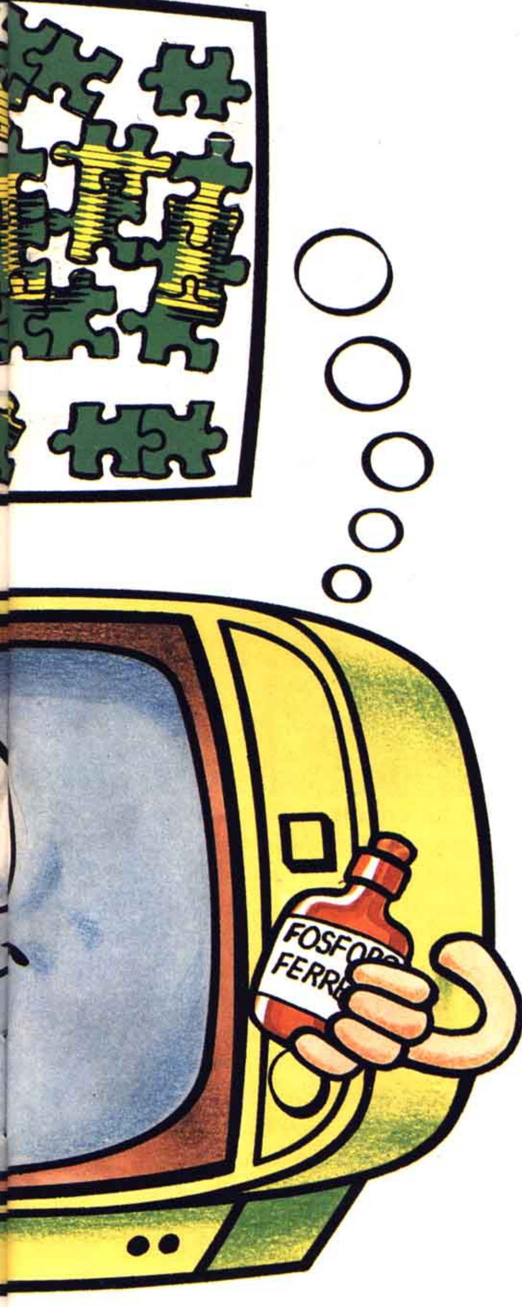


HOLA 32 72 79 76 65 32

Al dar la orden WIDTH 39 se ignora la columna de la izquierda de la pantalla (columna 0). Esta es la razón por la que el carácter que aparece en

el byte 0 es 32 (que corresponde al espacio en blanco: si en un cuadro no hay nada escrito, el byte corespondiente de VRAM contiene el valor 32).

Los códigos que hay en las posiciones 1 a 4 (72,79,76,67) son los de las



letras H,O,L y A, y por fin en el byte 5 vuelve a haber un 32 (espacio).

Probablemente ya hayas pensado que esto te permite escribir en los márgenes. Escribe ahora el siguiente programa:

```
10 SCREEN 0
20 WIDTH 20
30 LOCATE 0,0
40 PRINT"ANCHO=20"
50 FOR I%=1 TO 4
60 READ J%
70 VPOKE I%,J%
80 NEXT I%
90 DATA 72,79,76,65
```

Verás aparecer en la pantalla:

HOLA ANCHO = 20
Ok

Pero no podrás mover el cursor hasta la región donde está escrito «HOLA». Sin embargo si ahora tecleas CLS, sí se limpia toda la pantalla. Lo que hace CLS es llenar toda la tabla 0 de espacios, colocando el valor 32 en cada uno de sus bytes.

Otra cosa que podemos hacer con la tabla 0 es precisamente algo parecido a CLS pero sólo en un trozo de pantalla, por ejemplo, para dejar espacio para mensajes. Esto es similar a la técnica de las ventanas pero más simple. Supongamos que queremos «limpiar» sólo las 4 filas inferiores de la pantalla (para ello tendremos previamente que haber eliminado la presentación de las teclas de función haciendo uso de KEY OFF). Con el siguiente programa en BASIC puedes borrar desde el byte B% hasta el último de la pantalla. Hemos iniciado la rutina con la línea 60000 para que la puedas incluir en tus programas:

```
10 KEY OFF
20 B%=800
60000 FOR I%=B% TO 959
60010 VPOKE I%,32
60020 NEXT I%
```

También podemos hacer esto mediante una pequeña rutina en código máquina (ver rutina 1), que pasada a BASIC (por medio de la sentencia DATA de la línea 10.000) queda como sigue:

```
10 CLEAR 200,55000!
20 KEY OFF
30 FOR I%=1 TO 20
40 READ A$
50 POKE 55000!+I%,
     VAL("&H"+A$)
60 NEXT I%
70 DEFUSR=55001!
80 POKE 55099!,32
90 FOR I%=0 TO 959
100 VPOKE I%,65
110 NEXT I%
120 U=USR(800)
10000 DATA 3A,3B,D7,21,C0,03,
      ED,4B,F8,F7,ED,42,C5,E5
      ,C1,E1,CD,56,00,C9
```

El parámetro X de USR(X) (línea 120) debe ser el primer byte a borrar, y la rutina borrará desde este byte hasta el último de la tabla 0. Esta rutina llama a FILVRM que transfiere un valor a un trozo de VRAM.

El POKE de la línea 80 indica a la rutina en C.M. que tiene que llenar el trozo de pantalla con el caracter ESPACIO (código 32).

Prueba a llenar la pantalla con caracteres y a hacer uso posteriormente, de estas rutinas, tanto en BASIC como en código máquina.

Habrás observado que el byte correspondiente al cuadro (x,y) viene dado por

nº byte = y*40+x

Otra aplicación útil para la tabla 0 consiste en volcar toda la tabla en la memoria RAM, lo que puede ser útil, por ejemplo para presentar otras pantallas. La siguiente rutina en BASIC hace esto, transfiriendo desde el byte 0 al 959 (último de la tabla 0) de VRAM a los bytes 55100 a 56059 de RAM.

```
10 CLEAR 200,55000!
20 SCREEN 0
30 FOR I%=0 TO 959
40 VPOKE I%,65
50 NEXT I%
60000 FOR I%=0 TO 959
60010 POKE 55100!+I%,
      VPEEK(I%)
60020 NEXT I%
```

Para volver a pasar los datos de RAM a VRAM, puedes hacer uso de la siguiente rutina en BASIC

```
65000 CLS
65010 FOR I%=0 TO 959
```

```
65020 VPOKE I%,
      PEEK(55100!+I%)
65030 NEXT I%
```

Verás como al correr el primer programa se llena la pantalla con la letra A, y al correr el segundo, tras borrarse la pantalla, vuelve a llenarse con la letra A (por supuesto ignorando el WIDTH que en ese momento hayas fijado).

Las correspondientes rutinas en código máquina son las rutinas 2 y 3 que listamos en estas páginas. El siguente programa BASIC las incorpora mediante los DATAS de las líneas 10000 y 11000, respectivamente. Llamamos a estas rutinas mediante las funciones USR(0) y USR1(0), respectivamente (líneas 130 y 160). El bucle de la línea 150 es simplemente para poder apreciar el efecto de las rutinas.

```
10 CLEAR 200,55000!
20 KEY OFF
30 SCREEN 0
40 FOR I%=1 TO 26
50 READ A$
60 POKE 55000!+I%,
   VAL("&H"+A$)
70 NEXT I%
80 DEFUSR=55001!
90 DEFUSR1=55014!
100 FOR I=0 TO 959
110 VPOKE I,65
120 NEXT
130 U=USR(0)
140 CLS
150 FOR I=1 TO 100:BEEP:NEXT
160 U=USR1(0)
10000 DATA 21,00,00,01,c0,03,
      11,3c,d7,cd,59,00,c9
11000 DATA 21,3c,d7,01,c0,03,
      11,00,00,cd,5c,00,c9
```

Rápido ...¿verdad?

LA TABLA DEL GENERADOR DE PATRONES EN SCREEN 0

Ya sabes que si en una posición de la tabla 0 escribes el número 65, en la posición correspondiente de la pantalla aparecerá la letra A. Pero ¿cómo sabe el ordenador que lo que tiene que escribir es la letra A? En algún lugar deberá tener almacenado lo que debe escribir, es decir los caracteres ASCII. Pues bien, la tabla del generador de caracteres se ocupa de esto. Esta tabla tiene una longitud de 2048 bytes y puedes encontrar su dirección de comienzo mediante la variable BASE(2) que vale 2048. Si divides la tabla en grupos de 8 bytes, observarás que tiene 256 grupos, uno para cada caracter (figura 2). El grupo correspondiente a un código i comienza en la dirección:

código i = 2048+8*i

Así el grupo correspondiente a la letra A, cuyo código es el 65, comienza en:

2048+8*65 = 2568

y como tiene una longitud de 8 bytes termina en el byte 2575.

Veamos qué valores contienen estas posiciones. Escribe el siguiente programa:

```
10 FOR I%=2568 TO 2575
20 PRINT VPEEK (I%)
30 NEXT I%
```

Al ejecutarlo obtendrás los números 32,80,136,136,248,136,136,0. Si en lugar de la línea 20 escribes:

```
20 PRINT BIN$(VPEEK (I%))
```

Obtendrás:

```
00100000
01010000
10001000
10001000
11111000
10001000
10001000
00000000
```

O sea que hemos formado la letra A con ceros y unos. El ordenador, a la hora de presentar el caracter, ignora los ceros y dibuja un punto donde hay un uno. Por eso podemos ver la letra A como la vemos. Prueba a editar otros caracteres a partir de sus códigos ASCII.

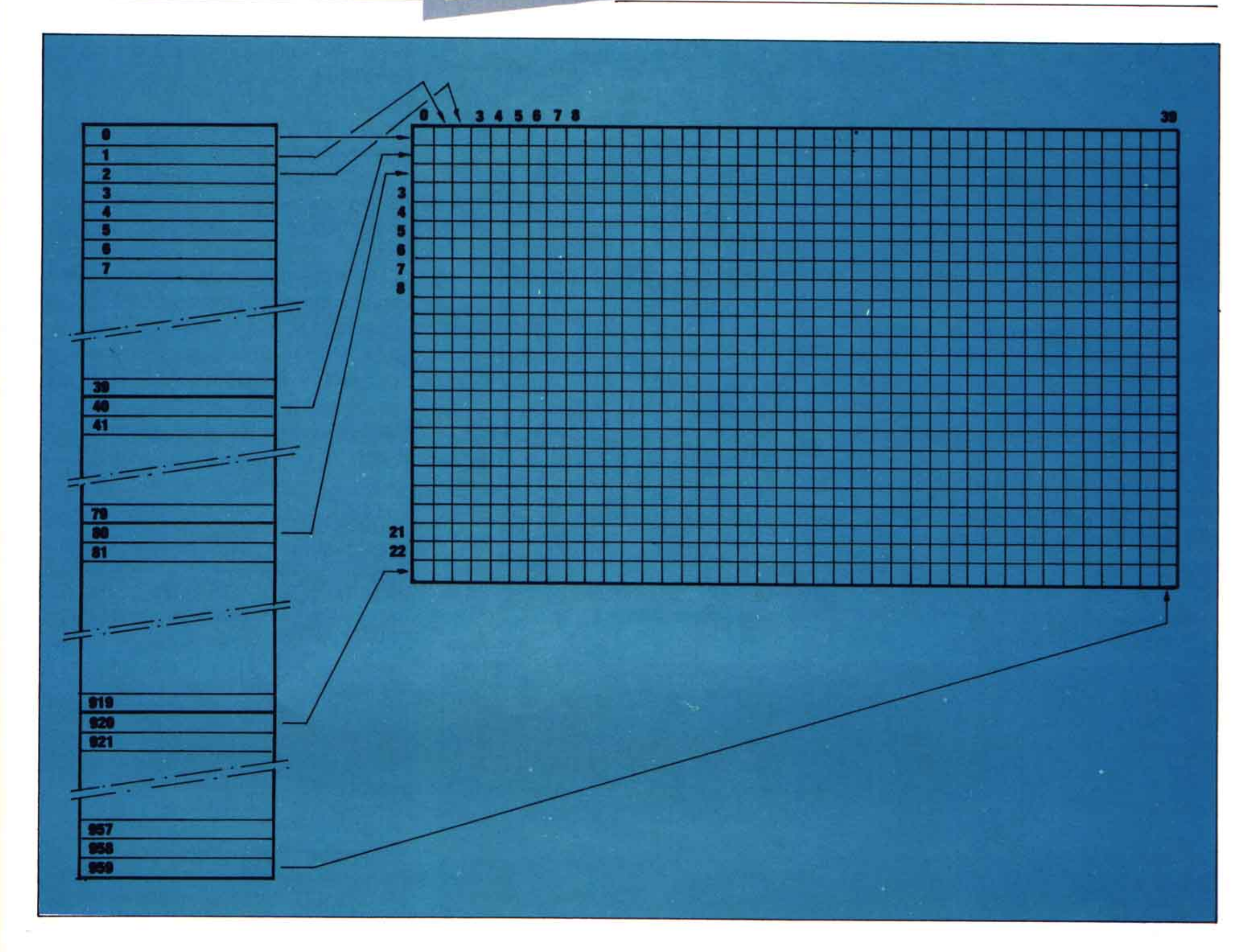
Al final del artículo incluimos un programa en BASIC que explicaremos entonces y que te permitirá editar todos los caracteres.

El hecho de tener una copia de los caracteres en un trozo de VRAM te hará pensar que pueden ser modificados. Efectivamente, observa (figura 2) que el último byte de la letra A contiene un 0. Lo mismo pasa con el resto de los caracteres, y esto es así para que haya una pequeña separación en la pantalla entre dos caracteres que están uno encima de otro. Escribe ahora VPOKE 2575,255 (255D = 11111111B). A partir de este momento cada vez que teclees la letra A mayúscula, ésta aparecerá subrayada.

Con el programa del final del artículo podrás modificar a tu antojo el juego de caracteres, y guardarlo en cinta o en disco. Esta posibilidad se incluye porque siempre que escribas SCREEN 0 el juego de caracteres se actualiza y vuelve a presentarse el juego inicial residente en ROM.

Como puede que te interese hemos incluido aquí un pequeño programa que te acepta dos juegos de caracteres que comprenden desde el caracter 0 hasta el 127 y sus inversos. Esto te permite escribir la letra A tal como se escribe normalmente y en negativo (el color de fondo pasa a ser el color de tinta y viceversa). Al utilizar este programa puedes escribir con caracteres normales hasta que pulses la tecla de función F1. A partir de entonces todo lo que escribas será en vídeo inverso, hasta que vuelvas a pulsar F1. La clave del programa está en la línea 30 en la que todos los caracteres por encima del 127 son modificados, de forma que en cada byte correspondiente al caracter de código I%+127 se escribe el «inverso» (complemento a 255) de lo que hay en el byte correspondiente del caracter de código I%, lo que se consigue con el XOR 255 de esa línea. Cuando ejecutes el programa espera un poco hasta que suene un BEEP ya que la línea 30 se ejecuta 1024 veces.

```
10 FOR I%=0 TO 127
20 FOR J%=0 TO 7
30 VPOKE 2048+(I%+128)*8+J%,
   VPEEK(2048+I%*8+J%)
   XOR 255
40 NEXT J%
50 NEXT I%
60 BEEP:BEEP
```




```
70 CLS
1000 ON KEY GOSUB 1050
1010 KEY(1) ON
1020 A$=INKEY$:IF A$=""
     THEN 1020
1030 PRINTA$;
1040 GOTO 1020
1050 ON KEY GOSUB 1000
1060 KEY(1) ON
1070 A$=INKEY$:IF A$=""
     THEN 1070
1080 PRINT CHR$(ASC(A$)+
     128);
1090 GOTO 1070
```

Con esto hemos terminado las tablas 0 y 2 que son todas las tablas disponibles en SCREEN 0. La tabla 0 tiene una longitud de 960 bytes y la tabla 2 de 2048, por tanto de los 16 Kbytes=16384 bytes de VRAM sólo se utilizan 3008 bytes. Quedan disponibles 13376 bytes que podemos utilizar para otras cosas, por ejemplo para guardar pantallas de ayuda. En un próximo artículo veremos como cambiar el valor de BASE(0) para tener hasta 14 pantallas diferentes.

LA TABLA DE NOMBRES DE PATRONES EN SCREEN 1

El modo de textos de 32 columnas, o SCREEN 1, poco se diferencia del de 40 columnas; al menos en principio. En este modo no disponemos de 40 columnas sino de 32, pero como esas 32 columnas ocupan también toda la anchura de la pantalla, cada uno de los cuadros correspondientes es más ancho que cada cuadro en el modo 0.

Figura 1.—Tabla de nombres de patrones en el modo de textos de 40 columnas.

Por eso en SCREEN 0 hay caracteres que no se ven completamente.

Podemos conocer la dirección de comienzo de esta tabla mediante la variable BASE (5)=6144, y como tiene sólo 32 × 24 cuadros (recuerda la figura 1: en lugar de 40 columnas, tenemos ahora 32), es decir 768 cuadros, su longiutd es precisamente esta (768 bytes). La organización de la tabla es análoga a la de la tabla 0. El byte correspondiente al cuadro (x,y) estará ahora en la dirección:

byte(x,y)=6144+32*y+x

Las aplicaciones de esta tabla son similares a las de la tabla 0. A continuación tienes la rutina que borra la pan-

talla desde el byte B% hasta el final de la pantalla.

```
10 KEY OFF
20 SCREEN 1
30 B%=640
60000 FOR I%=BASE(5)+B% TO
      BASE(5)+767
60010 VPOKE I%,32
60020 NEXT I%
```

Donde puedes modificar la variable B% para borrar una zona mayor o menor. La rutina correspondiente en código máquina es la rutina 4, a la que corresponde el siguiente programa BASIC:

```
10 CLEAR 200,55000!
20 KEY OFF
30 SCREEN 1
40 FOR I%=1 TO 20
50 READ A$
60 POKE 55000!+I%,
   VAL("&H"+A$)
70 NEXT I%
80 DEFUSR=55001!
90 POKE 55099!,32
100 FOR I=6144 TO 6911
110 VPOKE I,65
120 NEXT
130 U=USR(6784)
10000 DATA 3A,3B,D7,21,00,1b,
      ED,4B,F8,F7,ED,42,C5,E5
      ,C1,E1,CD,56,00,C9
```

Aquí también el parámetro X de USR(X) debe ser el primer byte a borrar, y la rutina borrará desde éste hasta el último de la tabla.

Para pasar toda la tabla de VRAM a RAM puedes utilizar el siguiente programa BASIC:

```
10 CLEAR 200,55000!
20 SCREEN 1
30 FOR I%=0 TO 767
40 VPOKE 6144+I%,65
50 NEXT I%
60000 FOR I%=0 TO 767
60010 POKE 55100!+I%,
      VPEEK(6144+I%)
60020 NEXT I%
```

En este programa (como ocurría con el correspondiente a SCREEN 0), la pantalla se llena con la letra A y luego la tabla se coloca entre las posiciones 55100 y 55867 de RAM. Para volver a pasar los datos de RAM a VRAM utiliza la siguiente rutina en BASIC:

```
65000 CLS
65010 FOR I%=0 TO 767
65020 VPOKE 6144+I%,PEEK
      (55100!+I%)
65030 NEXT I%
```

Las rutinas correspondientes en código máquina son las rutinas 5 y 6, que puedes ejecutar con el siguiente programa en BASIC:

```
10 CLEAR 200,55000!
20 KEY OFF
30 SCREEN 1
40 FOR I%=1 TO 26
50 READ A$
60 POKE 55000!+I%,VAL
   ("&H"+A$)
70 NEXT I%
80 DEFUSR=55001!
90 DEFUSR1=55014!
100 FOR I=6144 TO 6911
110 VPOKE I,65
120 NEXT
130 U=USR(0)
140 CLS
150 FOR I=1 TO 100:BEEP:NEXT
160 U=USR1(0)
10000 DATA 21,00,18,01,00,03,
      11,3c,d7,cd,59,00,c9
11000 DATA 21,3c,d7,01,00,03,
      11,00,18,cd,5c,00,c9
```

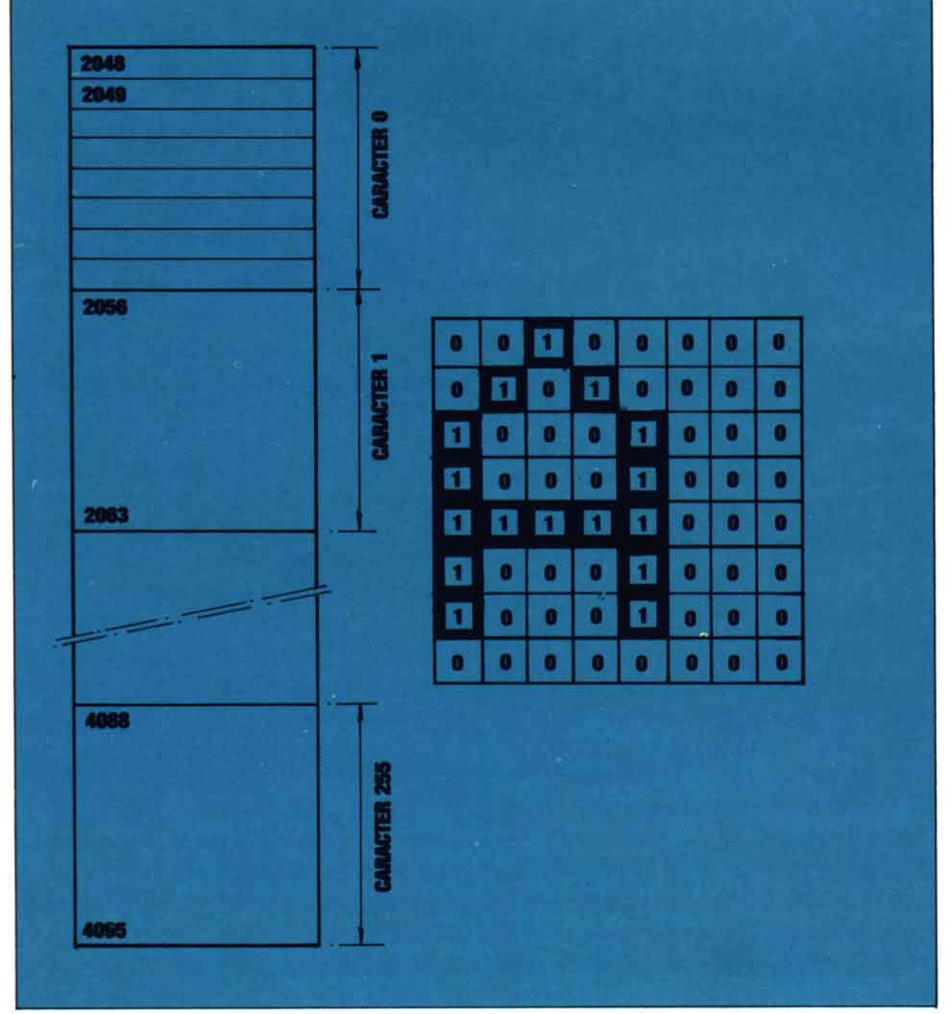


Figura 2.—Tabla del generador de patrones en el modo de texto de 40 columnas.

LA TABLA DEL GENERADOR DE PATRONES EN SCREEN 1

Esta tabla, cuya dirección de comienzo es:

BASE (7)=0

es en todo similar a la correspondiente tabla del modo 0, excepto, como ya has visto, en la dirección de comienzo. Ahora la dirección de comienzo del grupo i de 8 bytes correspondientes a un caracter viene dada por:

código i=8*i

Por tanto, el grupo correspondiente a la letra A comienza en SCREEN 1 en el byte:

8*65=520

y no en el 2568.

Seguramente serás capaz de modificar el programa que hemos dado anteriormente para disponer de dos juegos de caracteres en SCREEN 1.

LA TABLA DE COLOR EN EL MODO DE TEXTO DE 32 COLUMNAS

La principal diferencia entre los modos de pantalla 0 y 1 es que, mientras en SCREEN 0 sólo se puede definir un color para el fondo y un color para la tinta de todos los caracteres, en el modo 1 podemos definir distintos colores del fondo y de la tinta para cada grupo de 8 caracteres.

Así se puede definir un color del fondo y un color de la tinta para los 8 primeros caracteres, es decir los que tienen como códigos desde el 0 hasta el 7, ambos inclusive, otros colores del fondo y de la tinta para los caracteres 8 al 15,... y así sucesivamente. Como disponemos de 256 caracteres, estos se dividen en 32 grupos de 8, para cada uno de los cuales se puede definir el color del fondo y de la tinta.

Evidentemente en alguna parte de la memoria deberán estar almacenados estos colores. La tabla 6, que empieza en BASE(6)=8192, se ocupa de ello. Esta tabla tiene, por tanto, una longitud de 32 bytes: desde el 8192 hasta el 8223 ambos inclusive. El byte 8192 contiene los colores de los caracteres cuyos códigos son del 0 al 7, el byte 8193 desde el 8 hasta el 15, etc. Los colores se almacenan en la misma forma que en SCREEN 2. El *nibble* alto contiene el color de la tinta y el *nibble* bajo el color del fondo (figura 3). Inicialmente todos los bytes de esta tabla contienen el valor F4 (en hexadecimal) por lo que el color de la tinta es blanco (Fh=15d) y el del fondo azul (4h=4d) para todos los grupos de caracteres.

Además de estas tres tablas, en SCREEN 1 están las tablas necesarias para el uso de *sprites*.

Para terminar hemos incluido un programa BASIC bastante completo con el que puedes explorar los caracteres ASCII en los modos 0 y 1, modificar las tablas de generadores de patrones correspondientes a los modos 0 y 1 y modificar los colores de fondo, tinta y margen en ambos modos y los de cada juego de 8 caracteres en el modo 1.

Para explorar las tablas de generadores basta pulsar 0 y 1 según el modo que deseemos. En los dos modos podemos cambiar el caracter aumentando el código en 1, lo que se consigue pulsando F1 (es decir se comienza presentando el carácter 0, después el 1, luego el 2,...). Si en lugar de F1 pulsas F2 podrás introducir el código del carácter que desees ver, y si pulsas F3, basta con teclear el propio caracter (por ejemplo A).

Para modificar un caracter puedes actuar de la misma forma que para explorar la tabla. Acto seguido verás el cursor parpadeando sobre el carácter ampliado. Moviendo el cursor y pulsando la barra espaciadora puedes hacer que aparezca un punto de tinta donde no lo había o que se borre donde lo había.

Por último puedes modificar los colores de cada juego de caracteres en SCREEN 1 pulsando la tecla 5. Acto seguido te aparecerán 16 juegos de caracteres en grupos de 8. Para ver los 16 juegos restantes pulsa la barra espaciadora. Al pulsar F1 el programa te pedirá el número indicativo del juego cuyos colores quieres modificar y luego los nuevos colores, ¡ojo!, en hexadecimal.



Figura 3.—Composición de colores en la tabla de color en el modo de texto de 32 columnas.

En cualquier caso puedes volver al menú pulsando F4. También puedes grabar la tabla correspondiente en cinta o en disco. Recuerda que cuando cambias de modo de pantalla se restablecen las tablas originales, por lo que debes grabar cada tabla antes de abandonar el modo de pantalla correspondiente.

Si tienes alguna duda respecto a este programa (o respecto a cualquiera de las rutinas que incluimos) no dudes en escribirnos.

```
10 'EXPLORADOR DE VRAM EN
      MODOS 0 Y 1
20 '====================
30 '
40 'Juan Antonio Feberero
   Castejon
50 'Version 1.24.09.86 -
   5252 Bytes
60 '
90 CLEAR 200,55000!
100 ON STOP GOSUB 2770
110 STOP ON
120 P0%=4
130 T0%=15
140 P1%=4
150 F1%=4
160 SCREEN 0
170 WIDTH 39
180 LOCATE 3,11
190 PRINT "EXPLORADOR DE VRAM
    EN MODOS 0 Y 1"
200 PRINT "    ";STRING$(33,
    "=")
210 FOR I%=1 TO 2000:NEXT I%
220 SCREEN 0
230 COLOR T0%,P0%
240 KEY OFF
250 PRINT "0 = Explorar el
    modo 0"
```

```
260 PRINT "1 = Explorar el
    modo 1"
270 PRINT "2 = Modificar
    caracteres en modo 0"
280 PRINT "3 = Modificar
    caracteres en modo 1"
290 PRINT "4 = Modificar
    color en modo 0"
300 PRINT "5 = Modificar
    color en modo 1"
310 PRINT:PRINT:PRINT "Pulsa
    una opcion"
320 W$=INKEY$:IF W$=""
    THEN 320
330 IF INSTR("012345",W$)=0
    THEN 320
340 ON VAL(W$)+1 GOTO 350,450
    ,350,450,550,640
350 'Explorar modo 0
```

```
                    ;** RUTINA 1 **
                    ;
D6D9 3A3BD7              LD   A,(55099)
D6DC 21C003              LD   HL,960
D6DF ED4BF8F7            LD   BC,(0F7F8H)
D6E3 ED42                SBC  HL,BC
D6E5 C5                  PUSH BC
D6E6 E5                  PUSH HL
D6E7 C1                  POP  BC
D6E8 E1                  POP  HL
D6E9 CD5600              CALL 0056H
D6EC C9                  RET
                         END
```

```
360 '---------------
370 S%=0
380 GP%=2
390 W%=39
400 'Modificar caracteres
    modo 0
410 '--------------------
    ------
420 GOSUB 1150
430 NP%=0
440 GOTO 1650
450 'Explorar modo 1
460 '---------------
470 S%=1
480 GP%=7
490 W%=31
500 'Modificar caracteres
    modo 1
510 '--------------------
520 GOSUB 1150
530 NP%=5
540 GOTO 1650
550 'Modificar color modo 0
560 '----------------------
570 SCREEN 0
580 PRINT "Color actual de
    papel:";P0%
590 PRINT "Color actual de
    tinta:";T0%
600 PRINT
610 INPUT "(Nuevo color de
    papel";P0%
620 INPUT "(Nuevo color de
    tinta";T0%
630 GOTO 220
640 'Modificar color modo 1
650 '----------------------
660 SCREEN 1
670 WIDTH 31
680 COLOR ,P1%,F1%
690 PRINT "Color actual de
    papel:";P1%
700 PRINT "Color actual de
    fondo:";F1%
710 PRINT
720 PRINT "(Deseas cambiar
    colores de papel y de
    fondo ... S/N?"
730 W1$=INKEY$:IF W1$=""
    THEN 730
740 IF INSTR("Ss",W1$)=0
    THEN 780
750 INPUT "(Nuevo color de
    papel";P1%
760 INPUT "(Nuevo color de
    fondo";F1%
770 COLOR ,P1%,F1%
780 CLS
790 KEY1,"cambio"
800 FOR I%=2 TO 10
810 KEY I%,""
820 NEXT I%
830 KEY4,"menu"
840 KEY5,"grabar"
850 KEY ON
860 ON STRIG GOSUB 1100
870 STRIG(0) ON
```

```
                    ;** RUTINA 2 **
                    ;
D6D9 210000              LD   HL,0
D6DC 01C003              LD   BC,960
D6DF 113CD7              LD   DE,55100
D6E2 CD5900              CALL 0059H
D6E5 C9                  RET
                         END
```

```
                    ;** RUTINA 5 **
                    ;
D6D9 210018              LD   HL,6144
D6DC 010003              LD   BC,768
D6DF 113CD7              LD   DE,55100
D6E2 CD5900              CALL 0059H
D6E5 C9                  RET
                         END
```

```
880 ON KEY GOSUB 2280,,,2210,
    2400
890 KEY(1) ON
900 KEY(5) ON
910 I1%=0
920 FOR I%=I1% TO I1%+15
930 FOR J%=0 TO 7
940 IF I%*8+J%=127 THEN 980
950 IF I%<4 THEN C$=CHR$(1)+
    CHR$(I%*8+J%+64) ELSE C$=
    CHR$(I%*8+J%)
960 LOCATE J%,I%-I1%
970 PRINT C$
980 NEXT J%
990 LOCATE 9,I%-I1%
1000 PRINT USING"##";I%
```

```
1010 LOCATE 11,I%-I1%
1020 PRINT 8192+I%
1030 LOCATE 17,I%-I1%
1040 PRINT HEX$(VPEEK
     (8192+I%))
1050 NEXT I%
1060 LOCATE 0,18
1070 PRINT "(Codigo grupo?"
1080 PRINT "(Color (TF) en
     hexadecimal?"
1090 GOTO 1090
1100 'Pulsa barra espacios
1110 '--------------------
1120 I1%=I1% XOR 16
1130 CLS
1140 RETURN 920
1150 'Presentacion
     caracteres
1160 '------------
     ----------
1170 SCREEN S%
1180 WIDTH W%
1190 PRINT "C"digo="
1200 PRINT "----------"
1210 PRINT ".          ."
1220 PRINT ".          ."
1230 PRINT ".          ."
1240 PRINT ".          ."
1250 PRINT ".          ."
1260 PRINT ".          ."
1270 PRINT ".          ."
1280 PRINT ".          ."
1290 PRINT "----------"
1300 PRINT "Caracter:"
1310 KEY1,"1 a 1"
1320 KEY2,"codigo"
1330 KEY3,"tecla"
1340 KEY4,"menu"
1350 KEY5,"grabar"
1360 FOR I%=6 TO 10
1370 KEY I%,""
1380 NEXT I%
1390 KEY ON
1400 ON KEY GOSUB 1970,2010,
     2100,2210,2400
1410 I%=BASE(GP%)
1420 KEY(1) ON
1430 KEY(2) ON
1440 KEY(3) ON
1450 KEY(4) ON
1460 KEY(5) ON
1470 LOCATE 7,0
1480 I1%=I%§8-BASE(GP%)/8
1490 PRINT I1%
1500 FOR J%=0 TO 7
1510 A$=RIGHT$("00000000"+BIN
     $(VPEEK(I%+J%)),8)
1520 B$=""
1530 FOR K%=1 TO 8
1540 IF MID$(A$,K%,1)="0"
     THEN B$=B$+" " ELSE B$=
     B$+CHR$  (&HDB)
1550 NEXT K%
1560 LOCATE 1,J%+2
1570 PRINT B$;
1580 LOCATE 10
1590 PRINT I%+J%
1600 NEXT J%
1610 LOCATE 9,11
1620 IF I1%<32 THEN PRINT
     CHR$(1);CHR$(I1%+64)
     ELSE PRINT CHR$(I1%)
1630 IF INSTR("23",W$)<>0
     THEN RETURN ELSE RETURN
     1640
1640 GOTO 1640
1650 'Modificar caracteres
1660 '--------------------
1670 X%=1:Y%=2
1680 ON STRIG GOSUB 1910
1690 FOR I1%=1 TO 100:NEXT
1700 STRIG(0) ON
1710 LOCATE 0,12,0
1720 I3%=BASE(NP%)+Y%*(W%+1)+
     X%+1
1730 I4%=251 XOR VPEEK(I3%)
1740 FOR I1%=1 TO 100:NEXT
1750 LOCATE X%,Y%,1
1760 C%=STICK(0)
1770 STRIG(0) OFF
1780 IF C%=0 THEN 1690
1790 IF C%=1 THEN Y1%=1
1800 IF C%=3 THEN X1%=1
1810 IF C%=5 THEN Y1%=-1
1820 IF C%=7 THEN X1%=-1
1830 X%=X%+X1%
1840 Y%=Y%-Y1%
1850 X1%=0:Y1%=0
1860 IF X%<1 THEN X%=1
1870 IF X%>8 THEN X%=8
1880 IF Y%<2 THEN Y%=2
1890 IF Y%>9 THEN Y%=9
```

```
                  ;** RUTINA 3 **
                  ;
D6E6 213CD7              LD    HL,55100
D6E9 01C003              LD    BC,960
D6EC 110000              LD    DE,0
D6EF CD5C00              CALL  005CH
D6F2 C9                  RET
                         END
```

```
                  ;** RUTINA 6 **
                  ;
D6E6 213CD7              LD    HL,55100
D6E9 010003              LD    BC,768
D6EC 110018              LD    DE,6144
D6EF CD5C00              CALL  005CH
D6F2 C9                  RET
                         END
```

```
                  ;** RUTINA 4 **
                  ;
D6D9 3A3BD7              LD    A,(55099)
D6DC 21001B              LD    HL,1B00H
D6DF ED4BF8F7            LD    BC,(0F7F8H)
D6E3 ED42                SBC   HL,BC
D6E5 C5                  PUSH  BC
D6E6 E5                  PUSH  HL
D6E7 C1                  POP   BC
D6E8 E1                  POP   HL
D6E9 CD5600              CALL  0056H
D6EC C9                  RET
                         END
```

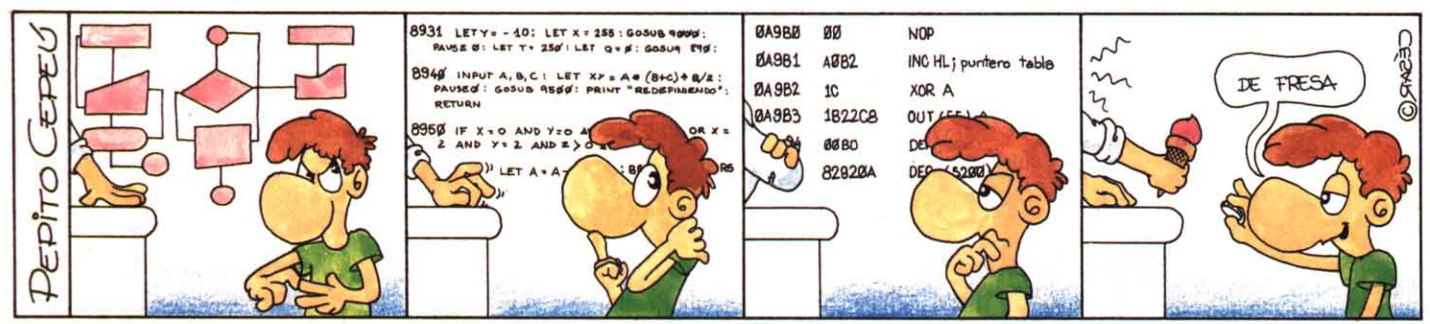
```
1900 GOTO 1690
1910 I1%=I%+Y%-2
1920 I2%=2^(8-X%)
1930 VPOKE I1%,VPEEK(I1%)
     XOR I2%
1940 LOCATE 0,12,0
1950 VPOKE I3%,I4%
1960 GOTO 1690
1970 'key 1
1980 '-----
1990 I%=I%+8
2000 GOTO 1420
2010 'key 2
2020 '-----
2030 DEFUSR=&H156
2040 U=USR(0)
2050 LOCATE 11,0
2060 PRINT STRING$(4,127);
2070 INPUT I%
2080 I%=I%*8+BASE(GP%)
2090 GOTO 1420
2100 'key 3
2110 '-----
2120 DEFUSR=&H156
2130 U=USR(0)
2140 LOCATE 10,11
2150 PRINT CHR$(127);
2160 I$=INPUT$(1)
2170 I%=ASC(I$)
2180 IF I%<32 THEN PRINT
     CHR$(1);CHR$(I%+64)
     ELSE PRINT CHR$(I%)
2190 I%=I%*8+BASE(GP%)
```

NO OLVIDES EL TELEFONO...

Cuando, por cualquier motivo, nos escribas.

```
2200 GOTO 1420
2210 'key 4
2220 '-----
2230 FOR I%=1 TO 5
2240 KEY(I%) OFF
2250 NEXT I%
2260 LOCATE,,0
2270 RETURN 220
2280 'key 1:color
2290 '-----------
2300 DEFUSR=&H156
2310 U=USR(0)
2320 LOCATE 17,18
2330 PRINT STRING$(4,127);
2340 INPUT C%
2350 LOCATE 30
2360 PRINT STRING$(4,127);
2370 INPUT C$
2380 VPOKE 8192+C%,VAL
     ("&h"+C$)
2390 RETURN
2400 'Grabar VRAM
2410 '-----------
2420 CLS
2430 PRINT "1 = Grabar en
     cinta."
2440 PRINT "2 = Grabar en
     disco."
2450 PRINT:PRINT:PRINT "Pulsa
     una opcion"
2460 W2$=INKEY$:IF W2$=""
     THEN 2460
2470 IF INSTR("12",W2$)=0
     THEN 2460
2480 IF W$="3" THEN 2540
2490 IF W$="5" THEN 2580
2500 RESTORE 2860
2510 IN%=BASE(2)
2520 L%=2143
2530 GOTO 2610
2540 RESTORE 2870
2550 IN%=BASE(7)
2560 L%=2143
2570 GOTO 2610
2580 RESTORE 2880
2590 IN%=8192
2600 L%=31
2610 IF W2$="2" THEN 2730
2620 FOR I%=1 TO 13
2630 READ W3$
2640 POKE 55000!+I%,VAL
     ("&H"+W3$)
2650 NEXT I%
2660 DEFUSR=55001!
2670 U=USR(0)
2680 PRINT:PRINT "Pulsa REC-
     PLAY ... <RETURN>"
2690 IF INKEY$<>CHR$(13)
     THEN 2690
2700 INPUT"(Nombre de archivo
     ";W3$
2710 BSAVE"cas:"+W3$,55014!,
     55014!+L%
2720 GOTO 2750
2730 INPUT"(Nombre de archivo
     ";W3$
2740 BSAVE W3$,IN%,IN%+L%,S
2750 PRINT:PRINT W3$;
     " grabado...<RETURN>"
2760 IF INKEY$<>CHR$(13)
     THEN 2760 ELSE 220
2770 'stop
2780 '----
2790 DEFUSR=&H3E
2800 U=USR(0)
2810 SCREEN 0
2820 WIDTH 39
2830 KEY ON
2840 LOCATE,,0
2850 END
2860 DATA 21,00,08,01,00,08,
     11,E6,D6,CD,59,00,C9
2870 DATA 21,00,00,01,00,08,
     11,E6,D6,CD,59,00,C9
2880 DATA 21,00,20,01,20,00,
     11,E6,D6,CD,59,00,C9
```





LOS OBJETOS DE LA AVENTURA

- DATA PARA LOS OBJETOS
- DESCRIPCIONES LARGAS Y CORTAS
- MAS VERBOS
- TOMANDO Y DEJANDO OBJETOS
- RUTINA DE INVENTARIO

Ha llegado el momento de llenar tu vacío mundo de aventura con objetos. Te enseñaremos cómo puedes incorporar en el programa tu lista de objetos y la forma de manejarlos.

Al final del capítulo anterior teníamos un conjunto completo de ambientes para la aventura y ya habíamos proporcionado al aventurero la capacidad de moverse por todo el mundo de la aventura. Sin embargo en la fase actual las actividades del aventurero todavía carecen de sentido, ya que aún no sucede nada en ninguno de los lugares. Ha llegado el momento de volver atrás y ver lo que habías planeado incluir en cada punto.

Vamos a ver a continuación, la forma de agregar las rutinas necesarias para llegar al sitio adecuado y recoger o abandonar todos los objetos que intervienen en la aventura. También presentaremos una rutina que hace un inventario de todos los objetos que el aventurero lleva consigo en un momento dado, la cual puede resultar útil en los momentos más críticos de la acción.

Carga el programa que tienes desde el capítulo anterior y prepárate a introducir nuevas rutinas.

OBJETOS

La máquina necesita saber tres cosas acerca de los objetos de una aventura: el número del lugar donde el objeto estaba inicialmente situado, listo para que el aventurero lo encontrara, un nombre para el objeto y una descripción más larga que incluirá algo acerca de la situación del objeto y sugerirá de alguna forma su empleo. Estas tres cosas son indispensables para el ordenador, ya que en primer lugar tiene que saber si un determinado objeto está o no en un ambiente, además



tiene que poder avisar al jugador de su presencia, con ayuda de la descripción larga y, por último, necesita un nombre corto, para utilizarlo en instrucciones e inventarios.

Los números de los lugares se situarán en una matriz, el título de los objetos en otra y las descripciones de los objetos en una tercera. El programa procesará las tres matrices en paralelo; cada elemento de la matriz soporta una información acerca de los objetos, el primero es el número del lugar, el segundo es el nombre, etc.

Añade a tu programa las siguientes líneas:

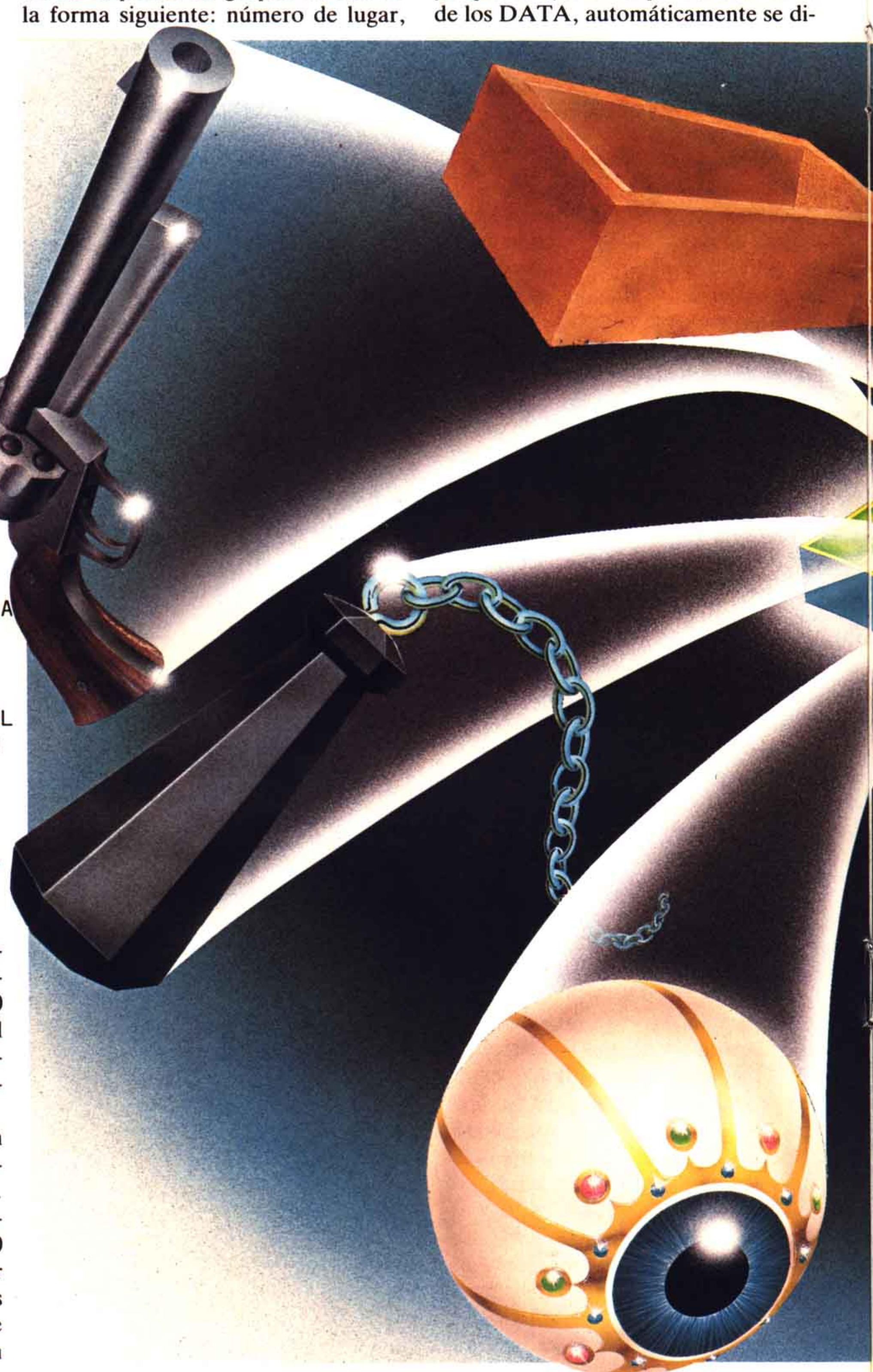
```
160 REM ** MATRICES OBJETOS
170 READ NB
180 DIM OB(NB),OB$(NB),
    SI$(NB)
190 FOR I=1 TO NB:READ
    OB(I), OB$(I),SI$(I):
    NEXT
200 DATA 7,4,BOLSA,HAY UNA
    BOLSA DE CANICAS
210 DATA 14,LADRILLO,HAY UN
    LADRILLO A TUS PIES
220 DATA 24,CADENA,UNA CADENA
    CUELGA DEL TECHO
230 DATA 0,PISTOLA,HAY UNA
    PISTOLA EN EL SUELO
240 DATA 0,GLOBO OCULAR,EN EL
    SUELO HAY UNA ALHAJA CON
    FORMA DE OJO
250 DATA 22,LAMPARA,VES UNA
    LAMPARA
260 DATA 0,INSPECTOR FISCAL,
    UN INSPECTOR FISCAL
    APARECE DE REPENTE
```

Cada línea desde la 200 a la 260 contiene tres elementos de los DATA referidos al mismo objeto. La línea 200 incluye un elemento suplementario: el número 7 al principio de los datos informa a la máquina de cuantos conjuntos de datos hay.

Una vez que se lee el número 7 en la línea 170, se dimensionan tres matrices con ese tamaño en la línea 180. OB contendrá la situación de cada objeto, ya sea un número de lugar, un 0 si el objeto todavía no existe (por ejemplo el famoso ojo de brillantes que tiene que ser descubierto durante la aventura), o un −1 si se trata de un objeto que el aventurero lleva consigo. OB$ contendrá las descripciones cortas y SI$ las descripciones largas.

La línea 190 llena las matrices con datos de las líneas 200 a 260. Los datos se disponen en grupos de tres de la forma siguiente: número de lugar, descripción corta del objeto y descripción larga del mismo.

Cuando uses esta rutina para otras aventuras, no tendrás que hacer muchas modificaciones en su estructura, ya que al ajustar el primer elemento de los DATA, automáticamente se di-



mensionarán tanto el bucle FOR ... NEXT como las matrices.

DISPOSICION DE LOS OBJETOS

A continuación el programa contiene toda la información referente a la naturaleza y colocación de los objetos. La siguiente rutina presenta la descripción larga del objeto en el lugar adecuado:

```
360 REM ** VISUALIZACION DE
    OBJETOS EN EL LUGAR
    APROPIADO
370 FOR I=1 TO NB:IF
    OB(I)=L THEN PRINT
    SI$(I)
380 NEXT I
```



En esta fase tienes que hacer una pequeña modificación en las líneas 330 y 340: cambia el GOTO 400 por GOTO 370. Las líneas 370 y 380 examinan la matriz que contiene los lugares de los objetos. Si alguno de los números de los lugares coincide con el del lugar donde se encuentra el objeto en ese momento —variable L— aparece la descripción larga del objeto a continuación de la del lugar. Esta rutina se puede usar sin modificaciones en otras aventuras.

MAS VERBOS

Ya tienes en tu aventura unos cuantos objetos esparcidos por los diversos lugares, pero como la máquina todavía no entiende más palabras que NORTE, SUR, ESTE y OESTE, el pobre aventurero no puede hacer nada con esos objetos. Imagínate la frustración de no poder coger esa apetitosa bolsa de canicas o no poder defenderte contra el inspector de hacienda. Por eso tienes que darle al ordenador un vocabulario de palabras que pueda reconocer, diciéndole qué debe hacer con los objetos. Más adelante veremos qué hacer si el jugador introduce una palabra que no esté en el vocabulario suministrado a la máquina.

Ya que el programa trata todas las palabras de las direcciones como verbos, el mejor sitio para los verbos que indican lo que hacer con los objetos será la matriz R$, y el mejor sitio para los correspondientes números será la variable R.

En consecuencia tendrás que hacer algunas modificaciones, empezando por la línea 130. Tienes que cambiar los límites del bucle FOR ... NEXT. La nueva versión de dicha línea es:

```
130 FOR K=1 TO 19:READ R$(K)
    ,R(K):NEXT
```

Seguidamente, añade las líneas 140 y 145:

```
140 DATA NADO,5,VACIO,6,LUZ,7
    ,FIN,8,LISTA,9,MATO,10,
    DISPARO,10,AYUDA,11
145 DATA COJO,2,RECOJO,2,
    LLEVO,2,PONGO,3,DEJO,3,
    ABANDONO,3,TIRO,4
```

Cada verbo tiene su correspondiente número. Verbos con el mismo número tienen el mismo significado, por lo que al ordenador se refiere, y realizarán la misma operación. Por ejemplo, programando las cosas para que el ordenador acepte COGER, TOMAR y LLEVAR, el aventurero se ahorrará mucho de gasto de tiempo innecesario intentando descubrir cuál de estas palabras tiene que usar. Puedes añadir con facilidad tus propias palabras en las líneas de DATA cambiando el bucle FOR ... NEXT de la línea 130 y poniendo los nuevos DATA al final de la línea 145. Tienes que hacer más modificaciones en otras partes del programa, pero ya te diremos más adelante lo que tienes que hacer.

OTRAS RUTINAS

Después de que has completado la lista de verbos con la última rutina, el ordenador necesita algunas rutinas que le permitan atender instrucciones tales como hacer que el aventurero lleve determinados objetos.

La subrutina que comienza en la línea 3010 define V$, N$ e I, que es un número extraído de la matriz R. Con esta corta rutina, la máquina podrá comprender el significado de cada respuesta del aventurero, según el valor de I.

```
500 REM ** ENCONTRAR OPCION
505 IF I=0 THEN GOTO 520
510 ON I GOTO 1010,1150,1240,
    1310,1410,1460,1500,1360,
    1080,1550,3110
520 PRINT:PRINT"NO CONOZCO...
    ";V$:GOTO 370
```

Cada uno de los números que figuran después de la sentencia ON ... GOTO en la línea 510 es el principio de una subrutina. Cada valor de I corresponde a un verbo o grupo de verbos diferente. Por ejemplo, si I = 10, se seleccionará la rutina «matar», es el décimo número de la línea, por lo que la rutina comienza en la línea 1550.

Si la subrutina de comprobación de instrucciones, que comienza en la línea 3010, no encuentra coincidencias para la parte V$ de R$, se asigna a I el valor 0. En tal caso no tendrá efecto la sentencia ON ... GOTO de la línea 510 y se presentará el mensaje de la línea 520.

LA TOMA DE OBJETOS

Ya tienes la rutina para el I = 1, correspondiente al caso en que el aventurero introduce una palabra de dirección; dicha rutina se encuentra en las líneas 1010 a 1060.

Cuando I = 2, significa que el aventurero ha tecleado una palabra de «coger», como COGER, TOMAR o LLEVAR. La siguiente rutina permitirá al aventurero llevarse consigo cualquier objeto que haya en el lugar en que se encuentra. Sería algo así:

```
1140 REM ** RUTINA COJO
1150 FOR G=1 TO NB
1160 IF N$=LEFT$(OB$(G),
     LEN(N$)) THEN 1190
1170 NEXT
1180 PRINT:PRINT"NO COMPRENDO
     ...";N$:GOTO 330
1190 IF OB(G)=-1 THEN PRINT:
     PRINT"YA LO TIENES":
     GOTO 330
1200 IF OB(G)<>L THEN PRINT:
     PRINT"NO ESTA AQUI":
     GOTO 330
1210 PRINT"OK":OB(G)=-1
1220 GOTO 330
```

En las líneas 1150 a 1170 se busca la matriz OB$ que contiene las descripciones cortas de objetos, para saber qué objeto es el designado por el aventurero. Si se encuentra el nombre del objeto, el programa salta a la línea 1190. Si el objeto no figura por ninguna parte de la aventura, la línea 1180 presenta el mensaje NO ENTIENDO, seguido del nombre de objeto tecleado por el aventurero.

Suponiendo que el objeto designado haya sido encontrado, hay que comprobar dos cosas. La línea 1190 examina el elemento de la matriz OB correspondiente a dicho objeto, para ver si ya está en poder del jugador. Si el jugador ya lo tiene (el correspondiente valor de la matriz es −1) se presentará el mensaje YA LO HAS COGIDO.

En la línea 1200 se comprueba si el objeto está presente, examinando nuevamente la matriz de lugares. Si no está presente, el programa dice: NO ESTA AQUI. Naturalmente, puedes cambiar estos mensajes por otros, si no se adaptan bien a tu aventura.

Si el objeto está en el mismo lugar que el aventurero y no ha sido cogido por éste, la línea 1210 dice OK y en el correspondiente elemento de la matriz de lugares se pone el valor −1.

ABANDONO DE OBJETOS

La rutina de «abandono» hace exactamente lo contrario que la anterior. Permite al aventurero dejar los objetos que no quiere llevar con él.

```
1230 REM ** RUTINA DEJO
1240 FOR G=1 TO NB
1250 IF N$=LEFT$(OB$(G),
     LEN(N$)) THEN 1270
1260 NEXT:PRINT:PRINT"NO
     COMPRENDO...";N$:GOTO
     330
1270 IF OB(G)<>-1 THEN PRINT:
     PRINT"NO LO LLEVAS":
     GOTO 330
1280 PRINT"OK":OB(G)=L
1290 GOTO 330
```

Esta rutina funciona de una forma muy parecida a la de «toma». Nuevamente se examina la matriz de descripciones cortas, esta vez se hace en las líneas 1240 a 1260. Si el objeto designado por el aventurero figura en la matriz, la línea 1270 comprueba si el aventurero lo lleva o no. Si no lo lleva, se presenta el mensaje NO LO LLEVAS.

Si el aventurero lleva el objeto, la línea 1280 envía el mensaje OK y se ajusta el correspondiente elemento en la matriz de situación de objetos OB.



P y R

¿Se puede utilizar un sintetizador de voz con una aventura?

Tu aventura podría resultar más interesante programando la máquina de modo que anuncie acústicamente los mensajes, direcciones y descripciones de objetos, en lugar de representarlos en la pantalla.

Consulta el manual de tu sintetizador para ver cómo se puede hacer que la máquina hable y sustituya a las instrucciones que contienen las sentencias PRINT.

Ahora tiene el mismo valor que el lugar actual, es decir L, en vez de −1 que significaba que lo llevaba el aventurero.

LA LISTA DEL BOTIN

Los aventureros desmemoriados estarán muy contentos de poder contar siempre que quieran con una lista de todos los objetos que llevan. Aquí tienes una rutina que hará exactamente eso:

```
1070 REM ** LISTA
1080 PRINT:PRINT"TIENES: ";
     :IN=0
1090 FOR G=1 TO NB
1100 IF OB(G)=-1 THEN PRINT
     TAB(10)OB$(G):IN=IN+1
1110 NEXT
1120 IF IN=0 THEN PRINT
     TAB(10) "NADA"
1130 GOTO 330
```

La línea 1080 envía el mensaje LLEVAS, seguido de la lista de objetos. El bucle FOR...NEXT comprueba todos los elementos de la matriz de situación de objetos. Esta vez los elementos importantes son los que tienen el valor −1, significando que el correspondiente objeto es uno de los que lleva el aventurero. Si el valor de un elemento es −1, se presenta la descripción corta del objeto en cuestión, tomándola de la matriz. El contador de inventario IN se incrementa en 1.

Si el aventurero no lleva ningún objeto, IN se queda a cero y la línea 1120 en vez de la lista de objetos presenta el mensaje NADA.

Las rutinas de «toma», «abandono» e «inventario» pueden utilizarse tal como están , ya que NB ha sido definido en una rutina anterior.

Almacena ahora el programa (SAVE), dejándolo dispuesto para recibir las rutinas finales que veremos la próxima vez. Se trata de las rutinas correspondientes al inspector de hacienda, el ladrillo, la lámpara, el encuentro de la joya, el final de la aventura y, finalmente, la instrucción que describe el objeto de la búsqueda.

Si ejecutas el programa ahora, verás que hay partes del mismo que funcionan mientras que en otras partes ocurren cosas extrañas. La razón es que todavía hacen falta unas cuantas rutinas y el programa salta a líneas que aún no existen.

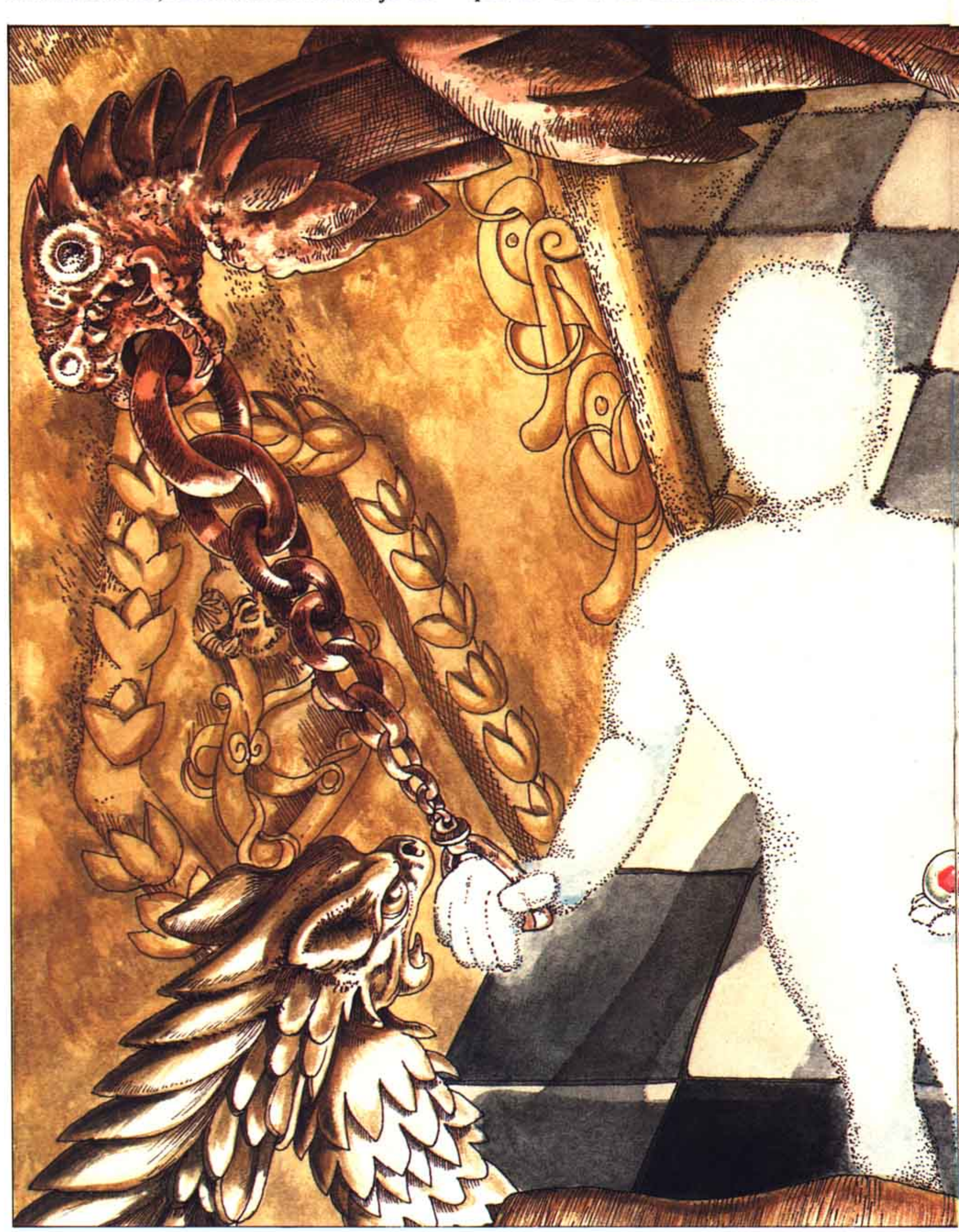
ELIMINANDO FALLOS

- Asegúrate de que los tres grupos DATA que contienen datos relacionados con los objetos se leen en la matriz adecuada. Si intentas meter datos de cadenas de caracteres en una matriz numérica, recibirás un mensaje de error, o puede que te encuentres con una descripción corta cuando esperabas una larga.

- Ten mucho cuidado en respetar el orden de los DATA en correspondencia con el orden de lectura de las matrices. El orden correcto es: lugar, nombre o descripción corta, descripción larga.

- Después de introducir los objetos haz una pasada de prueba del programa, para asegurarte de que los objetos aparecen en el sitio correcto.

- Al comprobar los objetos, hazlo sobre la retícula, para asegurarte de que no se te ha olvidado nada.



COMPLETANDO LA AVENTURA

- UNA RUTINA DE AYUDA
- EL INSPECTOR DE HACIENDA
- PROBLEMAS CON EL LADRILLO
- ENCENDIENDO LA LAMPARA
- INSTRUCCIONES

La aventura de INPUT **está ya casi completa. Sólo quedan las rutinas que la convierten en una aventura única, las rutinas especiales que confieren a este juego su carácter.**

Ya has completado casi todos los elementos que integran tu juego; es el momento de incorporar los últimos detalles del programa. Tienes que añadir aún algunas cosas, como los peligros, las advertencias, y tienes que prever un camino de salida para el caso de que la aventura se complete con éxito. Por último, tienes que poner las instrucciones para jugar.



Dado que muchas de estas rutinas están escritas de forma que contienen detalles específicos de una aventura particular, en los programas que siguen completaremos el juego y te mostraremos en términos generales todo lo que se relaciona con esta fase del programa. En otras aventuras no podrás utilizar estas rutinas sin modificarlas. En el próximo capítulo veremos la forma en que puedes adaptar estos principios para aplicarlos a tus ideas originales.

NECESITAS AYUDA

Si el juego de aventuras que has escrito es bueno, lo más probable es que el aventurero necesite alguna ayuda. Deberás pues prever unas cuantas sugerencias útiles.

Estas sugerencias tendrán la forma de mensajes que el ordenador enviará mediante sentencias PRINT en respuesta a las solicitudes de AYUDA hechas por el jugador. Lo que digan los mensajes y los puntos en que puedan estar disponibles, queda a tu entera discreción, ya que tú eres quien programa la aventura. Si quieres, puedes hacer que no haya un solo mensaje, o hacer que sean deliberadamente engañosos, o bien proporcionar ayuda únicamente en unas cuantas situaciones aisladas. El primer paso para decidir lo que incluir es volver a considerar tu plan original sobre las líneas maestras de la aventura.

En la aventura, hay varios puntos en los que convendría enviar al jugador un corto mensaje. Por ejemplo, podrías advertirle acerca de la habitación oscura, de forma que cuando esté en un lugar contiguo a dicha habitación, la respuesta ante su posible solicitud de ayuda podría ser un mensaje como: MIRA ANTES DE DAR EL SALTO, o incluso algo más críptico.

Otro sitio donde se podría incluir un mensaje de aviso es la orilla del río, donde, en el caso de que se decida a nadar, el aventurero corre el riesgo de ahogarse dependiendo de que lleve o no el ladrillo.

Naturalmente, puedes repasar una a una todas las situaciones en las que podría ser conveniente un poco de ayuda, pero supongamos que decides no ayudar demasiado y enviar un mensaje en un solo punto, el río. Tienes que hacer referencia a este número de lugar, el número 7, y a la variable que registra la presencia del ladrillo, OB(2):

```
3100 REM ** RUTINA AYUDA
3110 IF L<>7 OR OB(2)<>-1
     THEN PRINT:PRINT"LO
     SIENTO, NO TE PUEDO
     AYUDAR AQUI!":GOTO 330
3120 PRINT:PRINT"LOS
     LADRILLOS PESAN MUCHO Y
     HACEN QUE TE DUELA EL
     BRAZO":GOTO 330
```

Si el aventurero no está en la orilla (L<>7), o no lleva consigo el ladrillo (OB(2)<> −1), la línea 3110 presentará el mensaje LO SIENTO, NO TE PUEDO AYUDAR AQUI. Si al llegar a la orilla del río, el aventurero lleva el ladrillo y pide ayuda, la línea 3120 imprimirá el siguienten mensaje de advertencia: LOS LADRILLOS PESAN MUCHO Y HACEN QUE TE DUELA EL BRAZO.

Caso de que quieras hacerlo, no hay nada que te impida incluir una lista

completa de condiciones con sus correspondientes mensajes de aviso.

EL INSPECTOR DE HACIENDA

En la aventura interviene un sujeto que está por ahí merodeando, que es un inspector de hacienda el cual intenta recuperar parte de los impuestos impagados por el aventurero, confiscándole uno de los objetos que lleva. No se preocupa mucho de cuáles son esos objetos, por lo que en algunos casos incluso podría aceptar como pago un ladrillo.

Si el aventurero no tiene la suerte de llevar nada consigo en el momento en que se encuentra con el inspector de hacienda, será encerrado en una mazmorra en la que se pudrirá para siempre. Y aquí termina el juego.

El papel del inspector de hacienda es proporcionar al juego un elemento probabilístico, que resulte impredecible independientemente de cómo se encuentren las demás condiciones. Como en otros ejemplos de introducción de probabilidades en la programación, puedes hacerlo recurriendo a la función RND. Por lo demás, puedes tratarlo como otro objeto cualquiera; la única diferencia es que su situación no es fija, sino que se establece de forma aleatoria.

Aquí tienes las líneas suplementarias que añadir para que aparezca el inspector fiscal:

```
320 R=RND(-TIME):IF INT
    (RND(1)*15+1)=1 AND TA=0
    THEN OB(7)=L:TA=1
480 IF OB(7)=L AND I<>10
    THEN 1590
```

La línea 320 hace que el aventurero tenga una probabilidad sobre 15 de encontrarse con el inspector. Sólo se le permite aparecer una vez durante el juego, por lo que necesitas una variable, TA, para indicar si ha aparecido o no.

Si el número aleatorio es 1 o menor que 1/15, y el inspector todavía no ha aparecido, la línea 320 ajusta el valor que corresponde al inspector de hacienda en la matriz de situación de objetos. La presentación del mensaje del inspector se hace igual que si se tratase de un objeto, almacenándose en la descripción larga de la matriz.

La línea 480 tiene que ver con la supresión del inspector. Se limita a comprobar si has intentado matarle. Si no lo has intentado, el programa salta a la línea 1590.

LA RESPUESTA ANTE EL IMPUESTO

Cuando el inspector vuelve su fea cabeza, sólo hay una solución posible. El aventurero debe disparar contra él utilizando la pistola que se encontró en el río:

```
1540 REM ** RUTINA DISPARO
1550 IF OB(4)<>-1 THEN PRINT"
     CON QUE?":GOTO 320
1560 IF OB(7)<>L THEN PRINT
     V$;" A QUIEN?":GOTO 320
1570 PRINT:PRINT"MATASTE AL "
     ;OB$(7):OB(7)=0:
     GOTO 330
```

Esta rutina se utiliza cuando el aventurero escribe las palabras MATO o DISPARO. Si no lleva la pistola (OB(4) <> −1), la línea 1550 presentará el mensaje CON QUE? Análogamente si el inspector de hacienda no está presente y el jugador intenta matarle, la línea 1560 le pregunta A QUIEN?

La línea 1570 le dice al aventurero MATASTE AL INSPECTOR DE IMPUESTOS, y ajusta la matriz de situaciones de los objetos de forma que dicho inspector ya no existe.

LA VENGANZA DEL INSPECTOR

El aventurero se encuentra con lo siguiente:

```
1580 REM ** INSPECTOR FISCAL
1590 IN=0:OB(7)=0
1600 FOR K=1 TO NB
1610 IF OB(K)=-1 THEN
     IN=IN+1
1620 NEXT
1630 IF IN<>0 THEN 1640
1635 PRINT:PRINT"COMO NO
     LLEVAS NADA TE
     ENCIERRA "
1638 PRINT"EN UNA FRIA
     MAZMORRA":GOTO 1360
1640 R=RND(-TIME):K=INT
     (RND(1)*NB+1):IF OB(K)
     <>-1 THEN 1640
1650 PRINT "EL INSPECTOR TE
     CONFISCA EL OBJETO ";OB$
     (K):OB(K)=0:GOTO 330
```

Al inspector sólo se le permite aparecer una vez durante toda la aventura, por lo que la línea 1590 ajusta la matriz de situación de objetos poniendo a cero el que corresponde al inspector; OB(7). Esto no tiene efecto alguno en esta rutina, pero hace que una vez que salgamos de ella, el inspector no vuelva a aparecer. IN es un contador utilizado para comprobar si se llevan objetos.

En las líneas 1600 a 1620 se recorre toda la matriz de situaciones de objetos, para comprobar si cada uno de los objetos se lleva o no. Por cada objeto que se porte, se incrementa IN en uno.

Si el aventurero no lleva objeto alguno, el valor de IN permanece a cero y se presenta el siguiente mensaje: COMO NO LLEVAS NADA, TE ENCIERRA EN UNA FRIA MAZMORRA. El juego termina aquí, y se pregunta de nuevo al aventurero si quiere jugar otra vez saltando a la línea 1360.

Si por el contrario el aventurero lleva objetos consigo, en la línea 1640 se elige uno al azar. Si el número elegido corresponde a uno de los objetos transportados por el aventurero, dicho objeto es confiscado; si por el contrario dicho objeto no ha sido cogido, se selecciona otro número al azar, siguiendo así hasta que el número elegido corresponda a uno de los objetos transportados.

Después de que ha sido seleccionado un objeto, la línea 1650 informa al aventurero de que dicho objeto ha sido confiscado por el inspector. La matriz de situación de objetos queda modificada de forma que dicho objeto ya no existe.

LAS NUEVAS UNIDADES DE DISCO Y LA COMPATIBILIDAD

Las nuevas unidades de disco que incorporan casi todos los equipos de la segunda generación MSX2 **tienen una capacidad doble (720K) y graban en un formato diferente (doble cara) al de las primeras unidades de disco** MSX. **¿Hasta que punto son compatibles estos dos diferentes estándares?**



La diferencia fundamental entre estos dos tipos de unidad de discos está en que la grabación de información se lleva a cabo, en el caso de las unidades de simple cara, a través de un sólo cabezal de lectura/escritura, mientras que en las nuevas unidades se utilizan dos cabezales, uno para cada cara del *diskette*. De esta forma, las nuevas unidades son capaces de almacenar, en el mismo *diskette*, una cantidad doble de información (720 Kbytes frente a los 360 Kbytes de las unidades de simple cara).

Estas unidades no son exclusivas de los equipos de la segunda generación. Por ejemplo **Mitsubishi** comercializa la unidad **ML-03FD**, de doble cara, que pueden utilizarla tanto equipos de la primera como de la segunda generación.

Otro caso es el del equipo **SVI 738 X'press**, de **Spectravideo**, que siendo de la primera generación (aunque con características fuera de lo común) incluye unidades de doble cara.

Otro caso lo forman algunos equipos segunda generación como los primeros **Philips VG-8235**, en los que la unidad incorporada es de simple cara.

Para contemplar el panorama citemos los casos del **Sony HB-F500P**, del **Mitsubishi ML-G3** y de los nuevos **Philips NMS 8250** y **NMS 8280**. En todos ellos, y esta es la línea que seguirán probablemente todos los nuevos modelos **MSX**, la unidad de discos incorporada es de doble cara.

Con esta variedad de configuraciones lo primero que uno se pregunta es: ¿Qué pasa con la compatibilidad?

COMPATIBILIDAD ASCENDENTE

Al margen de que las nuevas unidades de disco graben en las dos caras del *diskette*, el resto de las características y de los formatos de grabación empleados son exactamente los mismos que los de las antiguas unidades de simple cara. En ambos casos, una cara del *diskette* queda formateada en 80 pistas de 9 sectores cada una y con capacidad para almacenar hasta 512 bytes en cada sector. Esta identidad de formatos y el hecho de que las unidades de doble cara comiencen a grabar información por la cara 1 y sólo pasen a la cara 2 cuando se han llenado los 360 Kbytes de la primera cara, garantiza la compatibilidad, siempre que la entendamos como compatibilidad ascendente.

El mes pasado, al hablar del nuevo equipo **ML-G3** de **Mitsubishi**, comentábamos que la unidad de *diskettes* incorporada, de doble cara, tenía un formato de grabación no compatible con el de las unidades de la primera generación (de simple cara) y que en ciertos casos iba a resultar necesaria una conversión de formatos. Esto va a ser cierto siempre que intentemos trabajar con los programas grabados en la cara 2 del **diskette,** utilizando una unidad de simple cara. Es físicamente imposible leer la información almacenada en la cara 2 si sólo tenemos un cabezal de lectura/escritura que trabaja en la cara 1.

Pero si hablamos de compatibilidad, hay que entenderla como compatibilidad ascendente, considerando que dos equipos son compatibles si los programas del equipo de menores prestaciones pueden funcionar sin problemas en el equipo de mayores prestaciones. En este caso hay que matizar y decir que la unidad de doble cara del **ML-G3**, y las del resto de los equipos **MSX2**, son perfectamente compatibles con las unidades de simple cara. Esto quiere decir que cualquier *diskette* grabado en una unidad de simple cara, podrá ser leído perfectamente por la unidad de doble cara. Los usuarios que asciendan desde una unidad de simple a otra de doble cara, podrán utilizar todo el *software* que tengan en *diskette*.

EL MAESTRO ZEN



ZEN **es el nombre de un paquete ensamblador, desensamblador, monitor producido para** MSX **por** Avalon Software **y disponible en dos versiones:** ***cassette*** **y** ***diskette*** **(en este último caso bajo el sistema operativo** MSX-DOS.

Aunque no es el programa más sofisticado para analizar o generar programas en ensamblador **Z80**, destaca por su sencillez de manejo y por permitir la creación de ficheros «.COM» (esto es, ficheros ejecutables que se incorporan al sistema operativo como nuevos comandos) cuando se trabaja desde **MSX-DOS**.

Una vez terminada la carga en memoria sabremos que estamos bajo el control del mismo cuando nos aparezca en sustitución del antiguo cursor «>» otro indicador compuesto por la palabra «ZEN>». A partir de este momento el programa espera que le ordenemos ejecutar cualquier comando.

El **ZEN**, para hacernos una idea, no es más que un pequeño editor de texto que dialoga con nosotros mediante unos determinados comandos. Si no entiende alguno de los que le introducimos nos responderá con el mensaje de error «HUH?». Si en vez de ordenarle algo presionamos la tecla de RETURN, se borrará la pantalla y el cursor se situará en la línea superior.

LOS COMANDOS

Para comunicarnos con el programa disponemos de 26 comandos. Estos son introducidos por la primera letra de su nombre, en inglés y en mayúsculas, seguido de un parámetro si fuese necesario.

La lista de comandos es la siguiente:

1.- A (Assemble)

Ensambla el programa fuente residente en memoria.

2.- B (Bye)

Se utiliza para salir del programa.

3.- C (Copy)

Sirve para mover un bloque de memoria.

4.- D (Down)

Desplaza el puntero del programa a la línea anterior.

5.- E (Enter)

Permite insertar líneas de programa en la posición que está señalada por el cursor.

6.- F (Fill)

Rellena una zona de memoria con el valor que queramos.

7.- G (Goto)

Ejecuta una rutina situada en memoria.

8.- H (Howbib)

Indica el comienzo y el final del programa fuente en memoria.

9.- I (In)

Informa sobre el estado de un *port* determinado.

10.- K (Kill)

Elimina el programa fuente de la memoria.

11.- L (Locate)

Permite buscar una cadena.

12.- M (Modify)



VS verifica programa fuente (ver. *cassette*).

VB verifica programa objeto (ver. *cassette*).

V visualiza directorio (ver. **MSX-DOS**).

22.- W (Write)

WS salva programa fuente.

WB salva programa objeto (ver. *cassette*).

WC salva programa objeto (ver. **MSX-DOS**).

23.- X (Xamine)

Muestra el contenido de todos los registros del **Z80**.

24.- Z (Zap)

Elimina n líneas desde la apuntada por el cursor.

25.- d (*disassemble*)

Desensambla un bloque de memoria.

26.- u (*unscramble*)

Desensambla una zona de memoria para que podamos dar un «vistazo rápido a su contenido.

Posibilita la modificación del contenido de la memoria.

13.- N (New)

Edita para su modificación la línea apuntada por el cursor.

14.- O (Out)

Envía un dato por el *port* especificado.

15.- P (Print)

Lista n líneas desde la señalada por el cursor.

16.- Q (Query)

Presenta en pantalla 64 bytes en formato hexadecimal y ASCII.

17.- R (Read)

RS lee programa fuente.

RB lee programa objeto (ver. *cassette*).

RC lee programa objeto (ver. **MSX-DOS**).

18.- S (Sort)

Visualiza en pantalla las etiquetas utilizadas durante el ensamblado.

19.- T (Target)

Posiciona el cursor en una línea determinada.

20.- U (Up)

Desplaza el puntero del programa a la línea siguiente.

21.- V (Verify)

ESCRIBIENDO PROGRAMAS FUENTE

El programa **ZEN** genera los programas fuente creando un pequeño fichero ASCII residente en memoria con ayuda de un editor de líneas. Estos programas fuente también pueden ser generados desde el **MSX-DOS** mediante la instrucción COPY CON NOMBRE.SRC o con cualquier otro editor de líneas que trabaje con formato ASCII, siempre que le añadamos la extensión .SRC.

El formato de cada sentencia escrita en ensamblador debe ser el siguiente:

ETIQUETA: CODIGO OPERANDOS; COMENTARIOS

Además de los códigos normales del **Z80** el **ZEN** añade otras pseudo-operaciones: DEFB o DB, DEFW o DW, DEFS o DS, END, LOAD y ORG. El uso de END es obligatorio al final de cada programa para indicarle al ensamblador donde termina su trabajo. También es importante no olvidarse de las instrucciones ORG xxxx para que el código objeto quede organizado en una dirección determianda y

LOAD xxxx para que el resultado de la compilación sea cargado en la dirección xxxx (las direcciones de ORG y LOAD no tienen por que coincidir).

Como el movimiento se demuestra andando vamos a trabajar con un ejemplo. Supongamos que tenemos la versión **MSX-DOS** y queremos teclear el siguiente programa:

```
; COMANDO EXTERNO
;
;     CLS
;
; PARA MSX-DOS
;
        ORG  2000H
        LOAD 2000H
CALSLT: EQU  001CH
CLS:    EQU  00C3H
        LD   IX,CLS
        LD   IY,0
        CALL CALSLT
        RET
        END
```

Sigamos los siguientes pasos:

```
ZEN>E
1 ORG  2000H
2 LOAD 2000H
3 CALSLT: EQU
4 CLS: EQU 00C3H
5 LD IX,CLS
6 LD IY,0
7 LD IX,CLS
8 RET
9.
```

Una vez tecleado esto corregimos los fallos.

Hemos duplicado una línea.

```
ZEN>T7
7 LD IX,CLS
ZEN>Z
```

Falta END al final del programa.

```
ZEN>T10
EOF
ZEN>E
8 END
9 .
```

La línea 3 está incompleta.

```
ZEN>T3
ZEN>N
3 CALSLT: EQU 001CH
```

Antes de ensamblarlo añadimos los comentarios para su posterior identificación y salvamos el programa fuente.

```
ZEN>T1
ZEN>E
1 ;
2 ; COMANDO EXTERNO
3 ;
4 ;       CLS
5 ;
6 ; PARA MSX-DOS
7 ;
8 .
ZEN>RS
NAME>CLS
```

Lo compilamos.

```
ZEN>A
OPTION>V
```

Ahora salvamos el programa fuente.

```
ZEN>WC
START>2000H
STOP>200AH
NAME>CLS
```

Salimos del programa y lo probamos.

```
ZEN>B
A>CLS
```

Esta rutina que además de muy corta es sencilla, puede ser de gran auyda para mejorar nuestros ficheros «.BAT». Sólo tendremos que tener la precaución de copiarla con el sistema operativo, ya que es un comando externo que le añadimos. Su hermano mayor **MS-DOS** ver 3.00 ya posee este comando como interno, pero, sólo se puede ver con la práctica.

EL ZOCO DE INPUT

Todo se compra y se vende. Los antiguos zocos fueron lugares destinados a todo tipo de transacciones. INPUT también tiene el suyo. Vuestras operaciones de compra, cambio o venta serán publicadas en esta sección, pero dos son las limitaciones que imponemos:

a) La propuesta tendrá que ver con la microinformática.
b) Nos reservamos el derecho de no publicar aquellos insertos de los que se sospeche un trasfondo lucrativo.

Ahora un ruego. Tratar de resumir al máximo el texto; escribir casi como un telegrama siendo claros y concisos.

Envía tu mensaje a:

INPUT MSX ZOCO
P.º de la Castellana, 93
28046 MADRID



5.25 PULGADAS EN TU MSX

Con la unidad SVI-707 **de** Spectravideo **y tu** MSX **podrás trabajar con *diskettes* del formato de 5.25 pulgadas y tendrás la oportunidad de intercambiar algunos ficheros con otros ordenadores que utilicen este formato, por ejemplo con todos los compatibles PC. ¡Interesante verdad!**

El formato elegido por los fabricantes de ordenadores **MSX** para las unidades de disco y los *diskettes* que iban a incorporar sus equipos fue, desde un primer momento, el de 3.5 pulgadas. Estas cifras, que hacen referencia al tamaño del *diskette*, corresponden a uno de los formatos que, más posibilidades tiene de convertirse en el estándar de los próximos años.

Este formato ha sido el resultado de una evolución, caracterizada por la búsqueda de un menor tamaño, de una mejor protección de la información y al mismo tiempo, de una mayor capacidad de almacenamiento de cada *diskette*.

El formato que primero se utilizó fue el de 8 pulgadas (más de dos veces mayor que el actual). Los *diskettes* de este formato eran demasiado grandes, difíciles de manejar y sin embargo no eran capaces de almacenar demasiada información.

El paso siguiente consistió en reducir el tamaño aproximadamente a la mitad y dió lugar a la aparición de los famosos *diskettes* de 5.25 pulgadas, que fueron los que incorporaron los primeros ordenadores personales, entre ellos el **Apple II**, los **Commodore** y los primeros **Atari**. También **IBM** eligió este formato para las unidades de disco de su línea de ordenadores PC.

Con ello las 5.25 pulgadas llegaron a convertirse rápidamente en el formato más popular para las unidades de disco de la mayoría de los ordenadores personales, liderazgo que se ha mantenido hasta nuestros días. Todavía hoy es el más popular de los formatos de *diskettes* y sobre el se ha grabado más información, en forma de programas y datos, de la que nunca se pudo imaginar.

¿Por qué entonces las 3.5 pulgadas para los *diskettes* del estándar **MSX**? La respuesta está en que este formato es el que mayores posibilidades tiene de convertirse en el estándar de los próximos años (desplazando a las 5.25 pulgadas) y ello porque supone una mejora en todos los aspectos. En primer lugar los *diskettes* son algo más pequeños, lo cual resulta interesante para los fabricantes, siempre preocupados por reducir el tamaño de sus equipos (sobre todo en el caso de los portátiles). En segundo lugar hay una importante mejora en cuanto a la protección de la información ya que los *diskettes* de 3.5 pulgadas tienen una funda rígida (en lugar de la funda flexible de los de 5.25) y una chapa metálica deslizante que oculta la superficie magnética del *diskette* cuando está fuera de la unidad (en los de 5.25 la superficie magnética queda al descubierto, expuesta a mil y un peligros). La última ventaja está en la mayor capacidad de almacenamiento del formato de 3.5 que actualmente se sitúa en torno a 1 Mbyte por *diskette* (en doble cara, doble densidad).

Estas razones, entre otras, han hecho que no sólo los fabricantes de **MSX** sino también firmas como **Commodore** (con su modelo **Amiga**) o **Atari** (en sus nuevos equipos **ST**) hayan optado por el estándar de 3.5 pulgadas. Hay incluso quien asegura que **IBM** está considerando adoptar los *diskettes* de 3.5 para sus nuevos modelos personales.

SVI-707

Spectravideo comercializa las 5.25 pulgadas en un conjunto formado por la unidad de *diskettes* **SVI-707**, un nutrido conjunto de manuales que incluye el de la propia unidad y los de los sistemas operativos **MSX-DOS**, **CP/M** y **MSX** Disk BASIC y varios *diskettes* de 5.25 pulgadas con los sistemas operativos mencionados y algunos programas de utilidad.

Sin embargo el conjunto no está completo, a menos que uno sea usuario del ordenador **SVI-728**, ya que el conector de la unidad de discos es específico para este ordenador. Para realizar la conexión a cualquier otro equipo **MSX** es necesario adquirir el cartucho adaptador que suministra la propia **Spectravideo** (a un precio de unas 2.000 pesetas) y que lleva el nombre de **SVI-213**. Este cartucho recibe por uno de sus extremos el conector de la unidad y por el otro encaja en los *slots* de cartucho de los **MSX**.

La unidad en sí, encerrada en una carcasa metálica de color blanco, es compacta y robusta. En su parte frontal, además de la ranura para introducir los *diskettes*, hay un par de diodos LED (indicadores luminosos) uno de ellos, de color verde, para indicar que la unidad está conectada a la red y el otro, de color rojo, para señalar que se está efectuando un acceso al *diskette* (en una operación de lectura o escritura).

La conexión a la red se efectúa a través de un alimentador externo, que, en nuestra opinión, hubiera sido mejor incluir en la carcasa de la unidad, pues ocupa espacio, estorba y añade cables extra a la maraña que siempre tenemos organizada.

LA UNIDAD EN ACCION

Una vez efectuadas las conexiones necesarias basta poner en marcha el ordenador para poder empezar a trabajar con la unidad. Su funcionamiento, cuando se utiliza como unidad única, no guarda sorpresas. Se comporta exactamente igual que las de 3.5 pulgadas. Lo primero que hicimos, tras formatear *diskettes* de 5.25 pulgadas, fue utilizar los comandos de **MSX-DOS**, **CP/M** y Disk BASIC para guardar y recuperar ficheros de varios tipos. No encontramos ni un sólo problema de funcionamiento, salvo cuan-



do probamos la versión **CP/M** en máquinas que no fueran **Spectravideo**. Al parecer la versión **CP/M** incluida sólo la soportan las máquinas de **Spectravideo** y no el resto de los **MSX**. Al trabajar con **MSX-DOS** y Disk BASIC no encontramos ningun problema de compatibilidad. Lo que si apreciamos, al hacer estos primeros ensayos, fue que la unidad es bastante silenciosa (algo muy importante) y que arranca algo más lentamente que las unidades de 3.5 pulgadas.

El paso siguiente consistió en conectar simultáneamente, dos unidades: la **SVI-707** y una unidad de 3.5 pulgadas. Nuestra intención era la de transferir ficheros de uno a otro formato de *diskette* y probar el funcionamiento de los programas después de las transferencias. En este caso, la unidad de 5.25 queda normalmente asignada como unidad A mientras que la de 3.5, según el tipo de conexión, queda como unidad C o E. De todos modos, tras una sencilla prueba queda claro cual es la letra correspondiente a cada una de las unidades.

Arrancamos con el sistema operativo **MSX-DOS** desde la unidad A y empezamos los ensayos. No hubo ni un solo problema. Transferimos todo tipo de ficheros de uno a otro formato de *diskette* sin que surgiera la menor dificultad. Hicimos las pruebas con distintos equipos, tanto de la primera como de la segunda generación **MSX**. Algunos equipos llevaban la unidad de 3.5 pulgadas incorporada y en otros casos había que utilizar una unidad externa (entonces conectabamos una unidad a cada uno de los dos *slots* de cartucho). Los ficheros copiados (mediante el comando COPY del **MSX-DOS**) ejecutaban perfectamente desde cualquiera de los dos formatos de *diskette*. En fin, las pruebas resultaron un completo éxito, y no hubo ni el más mínimo problema de compatibilidad, al menos con los equipos que probamos. No hicimos pruebas con todos los **MSX** existentes, así que no podemos garantizar que en algún caso determinado no vayan a surgir pegas. En cada caso concreto será conveniente asegurarse antes de que existe dicha incompatibilidad.

HACIENDO INTERCAMBIOS CON UN PC

El sigiente conjunto de pruebas que llevamos a cabo, y que para nosotros resultó el más interesante, estaba destinado a probar una de las afirmaciones más atractivas que se han hecho sobre el sistema operativo **MSX-DOS** que dice que el formato de grabación de ficheros de este sistema operativo es compatible con el de otro famoso sistema: el archiconocido **MS-DOS**.

MSX-DOS es un sistema operativo en disco fabricado por **Microsoft** (los mismos que han creado el **MSX-DOS**) que ha llegado a convertirse en el sis-



tema más popular de las máquinas de 16 bits al ser el de mayor uso en los ordenadores, **IBM** PC y compatibles.

Al lanzar el **MSX-DOS**, **Microsoft** anunció la compatibilidad de formatos con el **MS-DOS** lo que abría la puerta a una serie de interesantes manipulaciones y transferencias de ficheros entre máquinas tan distintas como un **MSX** y un PC.

El mayor obstáculo está en que los compatibles PC utilizan en general unidades de disco de 5.25 pulgadas, mientras que los **MSX** trabajan en 3.5. El disponer de la **SVI-707** nos iba a permitir hacer una conversión de formatos y probar de este modo la compatibilidad.

Desde luego que esta compatibilidad hay que entenderla como limitada a cietos tipos de ficheros, por ejemplo los ficheros ASCII, sin que en ningún momento pueda pensarse en la ejecución de programas de una máquina en otra, ya que tanto las CPUs, como los mapas de memoria e incluso los sistemas operativos, son totalmente diferentes.

Así que armados con nuestra **SVI-707** hicimos un primer intercambio e introdujimos un *diskette* de **MSX** en un PC y, al tiempo, uno de PC en un **MSX**. Tecleamos «dir» en ambas máquinas y pudimos ver como aparecían los directorios de cada *diskette* en una máquina que no era en la que habían sido grabados. La primera prueba había sido un éxito.

El siguiente paso fue utilizar dos procesadores de texto (la versión **Wordstar** en **MSX** y el programa **Personal Editor** en el PC) para intentar un intercambio de ficheros ASCII. Tras grabar unos textos de prueba intercambiamos los *diskettes* e intentamos la lectura. De nuevo el éxito fue completo. El texto grabado en el **MSX** aparecía intacto en el monitor del PC y viceversa. Con ello dábamos por terminado el experimento, con el que habíamos comprobado que, en efecto, los dos sistemas operativos de **Microsoft** graban en el mismo formato y permiten el intercambio de ciertos tipos de ficheros.

MUCHAS POSIBILIDADES

La unidad **SVI-707** es un sistema que permite la conversión entre los dos formatos más populares de unidades de disco y *diskettes* de la actualidad. Permite además las transferencias de ficheros entre los sistemas operativos **MSX-DOS** y **MS-DOS** de una forma sencilla y rápida.

Todo ello a un nivel de precios que se sitúa en torno a las 65.000 pesetas (incluyendo el precio de la unidad y del adaptador para cualquier **MSX**).

Las posibilidades que con todo ello se abren son inmumerables, especialmente si consideramos entornos profesionales en los que las conversiones de formato y la coexistencia de **MSX** y PCs, pueden llevar a soluciones muy económicas para instalaciones y ámbitos de programación complejos.

MITSUBISHI MAP

- ESTRUCTURA DE MAP
- TRATAMIENTO DE TEXTOS
- GESTION DE ARCHIVOS
- HOJA ELECTRONICA
- PROGRAMACION

El cartucho MAP es una interesante herramienta, con ella podrás realizar labores sencillas de tratamiento de textos, gestión de archivos, y presentación de gráficos. Todo ello desde un solo programa.

MAP se encuadra dentro de los paquetes integrados de programas de gestión que contienen fundamentalmente un procesador de textos, una base de datos, una hoja electrónica y una utilidad de gráficos. Todos ellos se organizan en torno a una base común de modo que puedan interaccionar fácilmente entre ellos.

ESTRUCTURA Y ORGANIZACION GENERAL DE LOS PROGRAMAS

MAP organiza la memoria del ordenador en cuatro zonas de trabajo, llamadas hojas: la de datos, la de formateo, la de programa y la de fichas.

La hoja de datos contiene la información que vamos a manejar. Esta información puede ser un texto, en el caso de que estemos trabajando con el procesador de texto, fichas, si utilizamos la base de datos, contenido de las celdas, en el caso de la hoja de cálculo, o valores con los que construir los gráficos. Este área puede contener hasta 100 líneas de información de 120 caracteres cada una.

La hoja de formato indica cómo se va a disponer la información en las líneas de la hoja de datos.

La hoja de programas se usa para almacenar los programas **MAP**, de los que hablaremos más adelante.

La hoja de fichas se usa como un *buffer* temporal para introducir la información en la hoja de datos.

Nosotros podremos introducir información en cada una de esas hojas por separado, así como guardar y recuperar la inforamción contenida en ellas, independientemente. Ello nos permite guardar, en *diskette* o en *cassette*, y en archivos independientes, la información de datos, de programas o de formato.

Hay que destacar que el programa nunca nos va a preguntar cual de las cuatro utilidades queremos usar. El uso del **MAP** como procesador de texto, base de datos, hoja de cálculo o generador de gráficos depende exclusivamente de la utilización que hagamos de los comandos, diferenciándose únicamente en la forma de disponer la información en la base de datos.

Dichos comandos pueden introducirse de dos maneras distintas. Puede hacerse uso de una tabla general, en la que aparecen todos los comandos, junto con un icono explicativo a cada uno. El comando se selecciona llevando el cursor a cada icono y pulsando RETURN.

La otra forma de introducir comandos es dando al ordenador el nombre del comando desde el teclado.

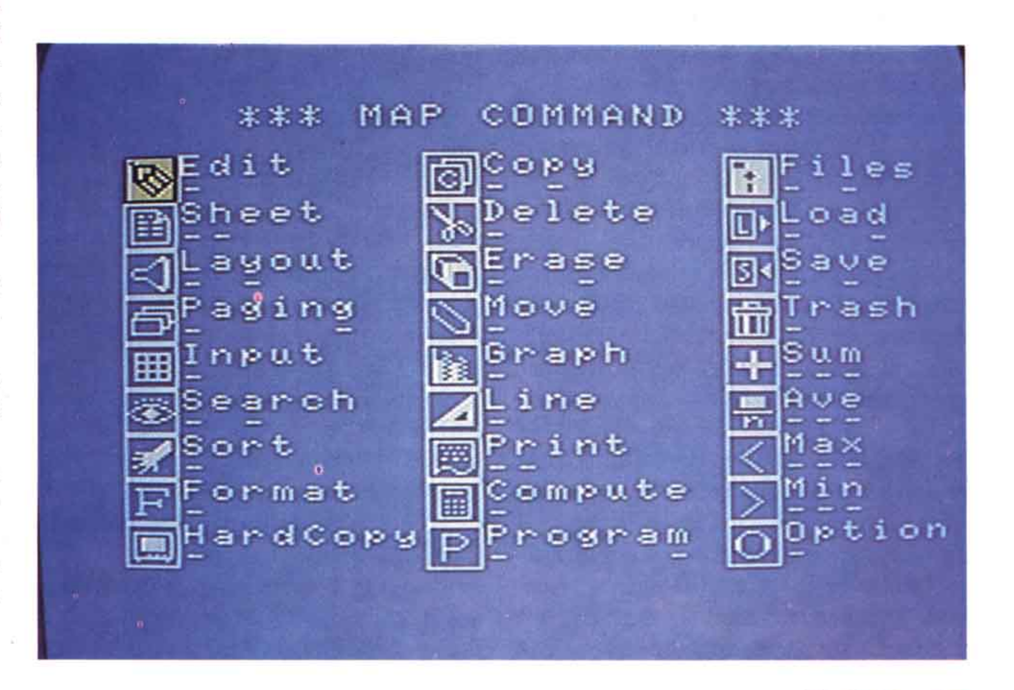
En cualquier momento podemos seleccionar uno de los dos modos anteriores, haciendo uso del comando OPTION. Este comando permite otras muchas selecciones, que más adelante veremos.

Vamos a estudiar el uso de **MAP** en cada una de las cuatro opciones mencionadas.

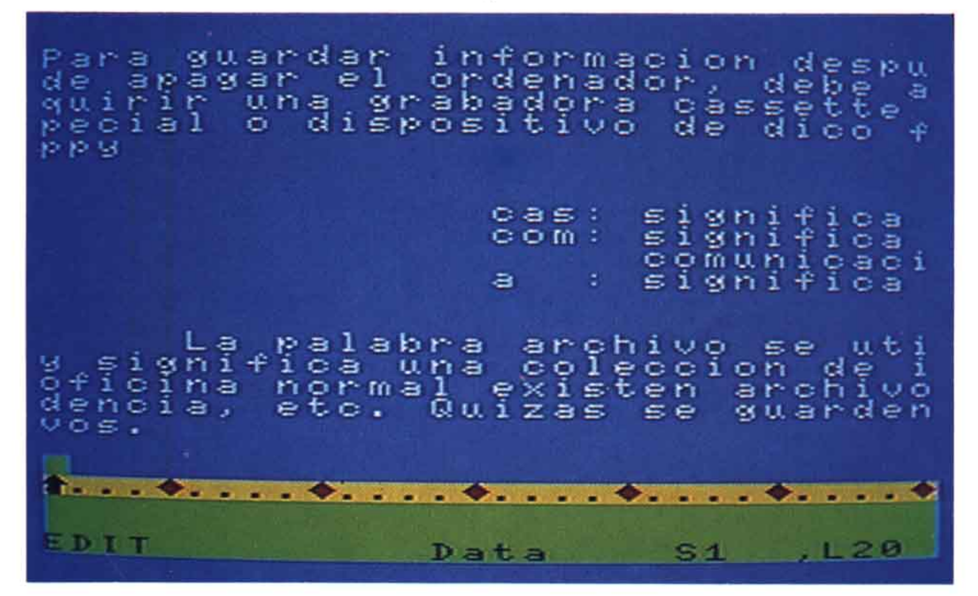
MAP COMO PROCESADOR DE TEXTOS

Usando **MAP** como procesador de texto, introduciremos directamente nuestra información en la hoja de datos. Esta información, corregidos todos los posibles errores y formateada a nuestro gusto, la volcaremos a la impresora.

MAP deja automáticamente 10 espacios en cada margen , por defecto. Podemos cambiar el tamaño de márgenes que deseemos haciendo uso del comando OPTION. También, y con este mismo comando estableceremos las divisiones de tabulación.



EL MENU PRINCIPAL DE MAP



MAP COMO TRATAMIENTO DE TEXTOS

Cuando escribamos nuestro texto en la hoja de datos tendremos la posibilidad de insertar caracteres y líneas, borrarlos, así como mover o copiar una porción de la hoja de datos en otra, y borrar zonas de texto.

La forma de mover, copiar y borrar información merece mención aparte, ya que la selección de la zona a modificar se efectúa dando la posición superior izquierda e inferior derecha del rectángulo de la hoja de datos que lo engloba.

Esta forma de marcar una zona en la hoja de datos para hacer algo en ella es común a todas las aplicaciones de **MAP**. No obstante, puede llegar a ser incómoda si el tamaño de la zona a mover es grande, pues obliga a moverse por la hoja de datos para marcar los límites.

MAP incluye también la posibilidad de hacer un *scroll* horizontal y vertical de la información en la hoja de datos, y puede ofrecer, sobre la pantalla, un esquema comprimido de dicha hoja, lo que nos permite localizar rápidamente donde está la información.

Deben resaltarse dos características importantes del uso de **MAP** como procesador de texto. Una de ellas es que, si lo deseamos, **MAP** evitará la división de palabras en el margen derecho pasando automáticamente la palabra a la línea siguiente si intentamos partirla. La otra consiste en la posibilidad de un alineado del texto en el margen derecho cambiando, si es necesario, el espaciado entre palabras. Estas dos características son seleccionables a través del comando OPTION.

MAP COMO BASE DE DATOS

Así como antes utilizabamos la hoja de datos para guardar el texto a procesar, ahora va a contener los elementos de información de la base de datos.

Básicamente, cada elemento de información (o ficha) los guardaremos en una de las líneas de la hoja de datos. Disponemos así de 120 caracteres por ficha. Esto puede ser poco en algunas aplicaciones.

Previamente al uso de la base de datos, tenemos que decirle a **MAP** cómo queremos organizar la información dentro de la línea o ficha; es decir, cuantos campos tiene cada ficha y cuantos caracteres tienen cada campo. Todo esto se realiza mediante el comando FORMAT.

Seguidamente introduciremos la información de las fichas en la base de datos. Esto se puede hacer ficha a ficha, con el comando PAGING. Se pedirá la información según lo indicado mediante el comando FORMAT. Sin embargo (y esto es general a todas las aplicaciones de **MAP**) hay otra manera de suministrar la información y es editando directamente la hoja de datos, mediante el comando EDIT. Caso de escoger este camino, es responsabilidad nuestra teclear la información en su lugar adecuado.

Las facilidades de **MAP** como base de datos son las usuales en este tipo de aplicaciones, e incluyen el cambio y acutalización de la información, la localización de una ficha o grupo de fichas determinado, y la ordenación alfabética o por valor numérico.

Siempre podremos guardar la información de las hojas de datos y de formato en *diskette* o *cassette* y recuperarla más adelante, así como imprimir las fichas de la base de datos.

MAP COMO HOJA DE CALCULO

En esta aplicación la información contenida en la hoja de datos se organiza en celdas, seleccionables por fila y columna, cada una de las cuales puede contener un número o un valor alfanumérico (como indicador del tipo de información en la fila o la columna).

Análogamente a como hacíamos con la base de datos, tendremos que instruir a **MAP** sobre la forma en que queremos organizar la hoja de cálculo. Esto lo hacemos de nuevo mediante el comando FORMAT. Con él dividiremos los 120 caracteres de cada línea en la hoja de datos en tantas celdas como unidades de información necesitemos.

Existen tres formas básicas de introducir información en la hoja electrónica. La primera de ellas es de nuevo mediante el comando PAGING. En realidad, no existe diferencia entre la forma de tratar la información de la base de datos y de la hoja electrónica, siguiendo la filosofía general de **MAP**.

Otra forma es, de nuevo, mediante el comando EDIT, accediendo directamente a la hoja de datos. Esta forma es útil para dar título a toda una fila o columna.

La tercera forma es con el comando INPUT, con el que podremos suministrar los datos de entrada directamente sobre las celdas de la hoja de cálculo que seleccionemos. La forma de hacer la selección es la tradicional

en **MAP**: marcando los límites superior izquierdo e inferior derecho de la zona.

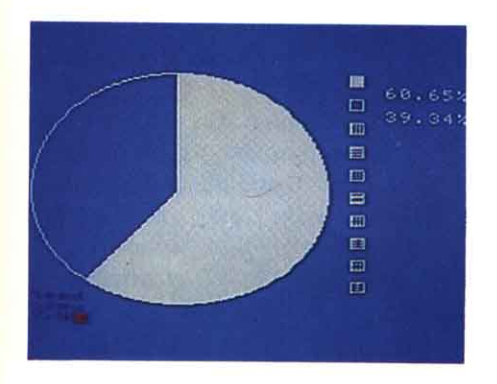
En cuanto a las facilidades que ofrece **MAP** como hoja electrónica se pueden cambiar y actualizar los contenidos de cada celda, borrar columnas y líneas (lo que obliga a actualizar el formato), y realizar cálculos entre elementos de celdas, filas o columnas.

MAP tiene además implementadas las siguientes funciones:

SUM: Encuentra la suma de todos los elementos de una fila o columna.

AVE: Promedia todos los números de una fila o columna.

MAX: Encuentra el máximo de una fila o columna.



MIN: Encuentra el mínimo de todos los elementos de una fila o columna.

GRAFICOS MAP

MAP permite la presentación automática de los datos numéricos existentes en la hoja de datos en forma gráfica. Se pueden elegir tres tipos de representación: diagrama de barras, diagrama de líneas y diagrama de sectores. Además, los diagrams de barras y de líneas permiten comparar hasta 5 magnitudes simultáneamente, representándose por valores y puntos distintos.

Los números se introducen en la hoja de datos por cualquiera de los procedimientos ya comentados. Una vez allí, el comando GRAPH trazará el tipo de gráfico seleccionado en la pantalla.

GRAPH nos preguntará qué zona de la hoja de datos vamos a representar, y si deberá tomar la información por filas o por columnas.

En los gráficos de líneas y de barras nos permitirá escoger la escala a utilizar y el punto de comienzo.

Los gráficos **MAP** son de muy buena calidad y podremos sacarlos por impresora, para incluirlos en hojas de resúmenes, demostraciones, etc.

PROGRAMAS MAP

Si vamos a gestionar la inforamción siempre de la misma forma pero utilizando distintos conjuntos de datos, existe la posibilidad de confeccionar un programa **MAP** y almacenarlo en cinta o *diskette*, para poder utilizarlo cada vez que deseemos.

Para ello usaremos el comando EDIT, escribiendo nuestro programa en la hoja de programas de **MAP**.

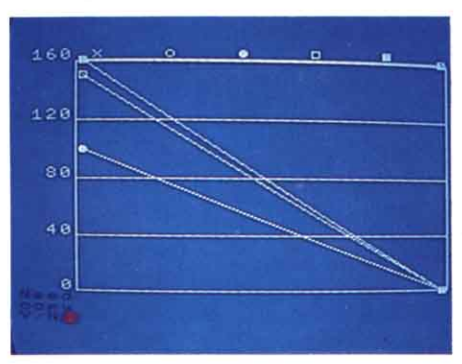
Los programas, además de los comandos, permiten instrucciones especiales, tales como:

Sacar información por pantalla (DISPLAY)

Introducir información desde el teclado (ACCEPT)

Emitir sonido (BEEP)

GRAFICOS MAP



Tenemos dos instrucciones especiales que permiten alterar el flujo de control. Estas instrucciones son:

GOTO etiqueta

IF expresión 1 = expresión 2 THEN comando

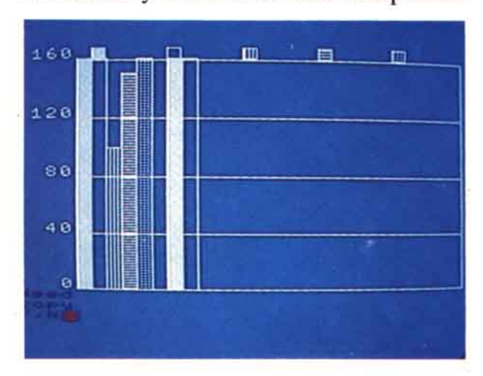
Como vemos, **MAP** no nos ofrece un control muy amplio sobre el flujo de los programas. Sin embargo, estas instrucciones serán suficientes para la mayoría de nuestros propósitos.

La limitación de la hoja de programas es de 60 líneas, lo cual es de todo punto escaso para programas un poco complicados o largos. Esto es una limitación seria para las capacidades de **MAP**.

ASI ES MAP

MAP es un paquete integrado de gestión de uso simple, sin pretensiones. El juego de comandos que ofrece es reducido y general, es decir, no depende del tipo de aplicación que estemos utilizando en ese momento, y todos los comandos valen en todas las aplicaciones.

Sin embargo, esta simplicidad de uso puede hacer que el manejo sea a veces engorroso. Se echan en falta funciones y comandos más completos.



Es posible que el afán por uniformizar la información a tratar, que es clave en **MAP**, contribuya a dificultar un tanto su manejo. Podrían haberse incluido funciones más orientadas a cada aplicación.

A nuestro juicio, se echa en falta la posibilidad de una interacción de la información con el dispositivo de almacenamiento (típicamente *diskette*) para no tener que limitar la capacidad de **MAP** a la memoria física del ordenador. En muchas aplicaciones, ésta puede ser insuficiente y obligarnos a procesar la información por trozos en el *cassette* o en el disco, lo que puede resultar tedioso.

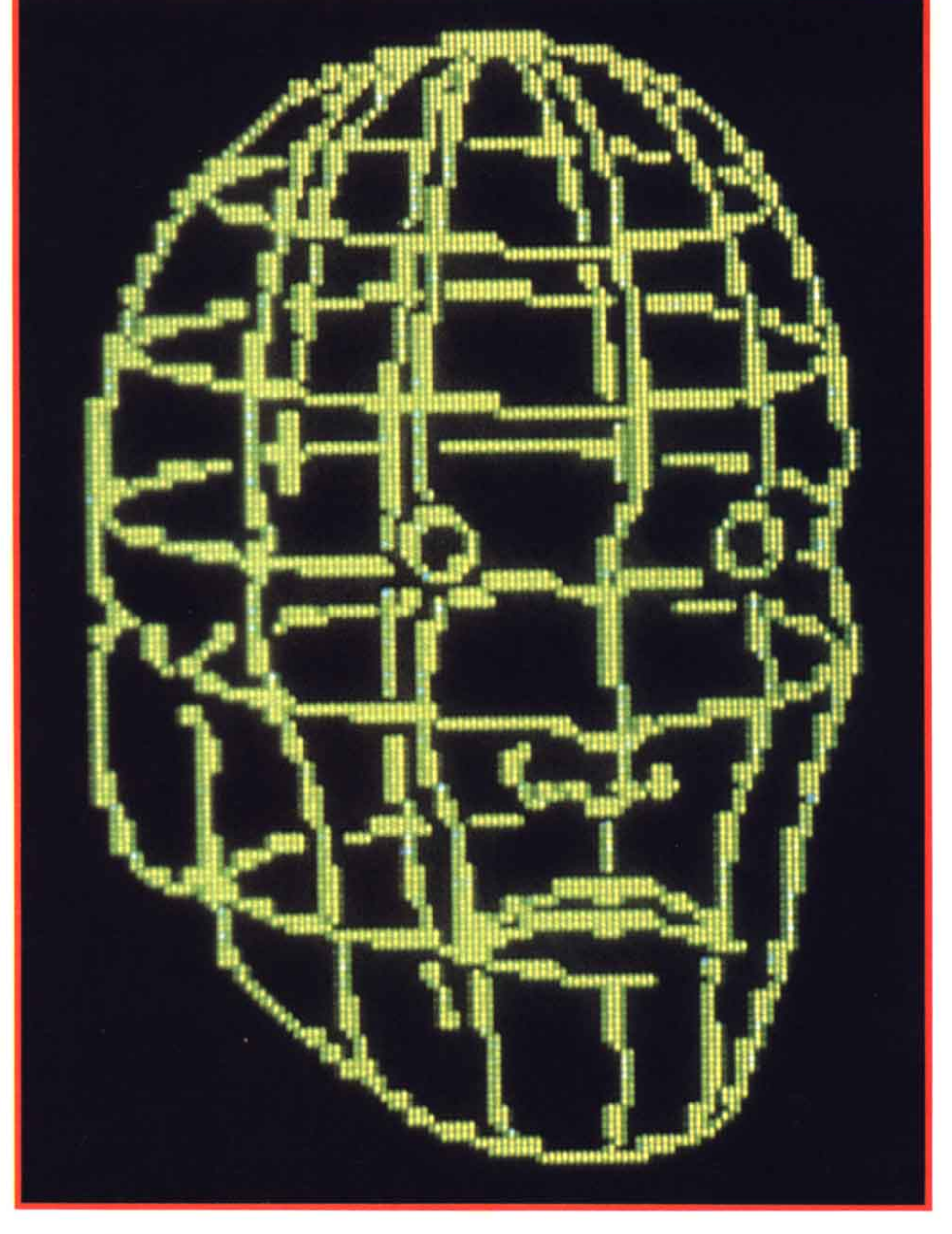
No queremos desmerecer, sin embargo, con esto, las capacidades de **MAP** que, con muy poco entrenamiento, se puede convertir en un valioso ayudante para la confección de trabajos, de presupuestos, gestión de pequeños negocios, preparación de demostraciones que incluyan gráficos, y un sinfin de aplicaciones más.

UN PROGRAMA QUE JUEGA A LAS CUATRO EN RAYA (I)

- JUEGOS DE AJEDREZ
- CRITERIOS DE EVALUACION
- ARBOLES DE DECISION
- LAS CUATRO EN RAYA
- EL PROGRAMA

En esta ocasión, vamos a hablar de una de las manifestaciones más apasionantes de la Inteligencia Artificial**: los juegos de estrategia. En primer lugar describiremos sus fundamentos, para pasar después a profundizar en su forma concreta de funcionamiento a partir de varios ejemplos cercanos a nosotros, con el juego de las «tres en raya», o el ajedrez.**

Aunque el ajedrez, como veremos más adelante, es uno de los ejemplos menos didácticos que podíamos proponer, sí es de los más interesantes, y por ello vamos a empezar por él.



El programa jugador de ajedrez más rápido del mundo (también podríamos decir que «el mejor»), es el llamado **Belle**, capaz de estudiar en torno a las 160.000 posiciones por segundo y simular sus jugadas y las respuestas del adversario en una profundidad de diecisiete niveles. No obstante, tan espectacular despliegue de medios no llega a ser suficiente para derrotar a los grandes maestros, si bien algunos de ellos han reconocido que al menos lo han tenido difícil para vencer.

El ajedrez, como todos los juegos de estrategia, requiere que los jugadores vayan mucho más allá del mero conocimiento de las reglas y la confianza ciega en la suerte; es necesario tratar en todo momento de maximizar el valor de nuestra posición frente al adversario, limitando su libertad de acción, coartando su iniciativa, y colocándole en una situación desfavorable que se haga susceptible de llevarle a la derrota. Ciertamente es muy difícil conseguir que una máquina pueda hacer todo esto mejor que el hombre.

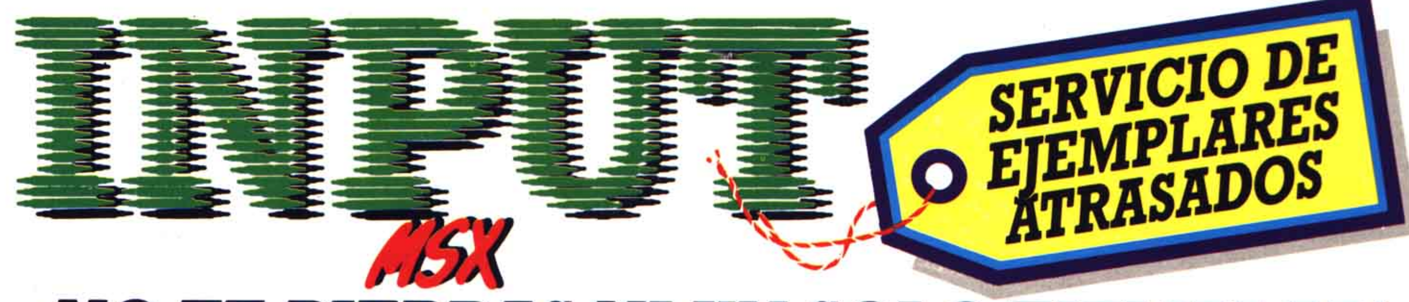
El único camino posible, que se sepa hasta ahora, necesita cubrir dos fundamentos básicos:

1º.- Establecer una serie de «criterios de evaluación» (cuantos más, mejor), que permitan al programa valorar su situación y saber así qué jugada es favorable y qué jugada no lo es.

2º.- Prever sus posibles jugadas remontándose en un árbol de posibilidades, valorando los resultados.

JUEGOS INTELIGENTES

La **Inteligencia Artificial** puede llegar a hacernos creer que comprende conceptos, pero no debemos engañarnos: el ordenador sólo comprende cifras. Por ello, vamos a hablarle en su idioma, el idioma de los números.



¡NO TE PIERDAS NI UN SOLO EJEMPLAR!

INPUT MSX quiere proporcionar a sus lectores este nuevo servicio de ejemplares atrasados para que no pierdan la oportunidad de tener en sus hogares todos los ejemplares de esta revista, líder en el mercado español.

Podréis solicitar cualquier número de **INPUT MSX** que querais, siempre al precio de cubierta (sin más gastos).

Utiliza el cupón adjunto, enviándolo a **EDISA** (Dpto. de Suscripciones), López de Hoyos, 141 - 28002 Madrid, o bien llámanos por teléfono al (91) 415 97 12.







CUPON DE PEDIDO

SI, envíenme contrarreembolso .... ejemplares de **INPUT MSX** de los números:

(marca con una (X) tu elección)

[1] [2] [3] [4] [5] [6]

NOMBRE
APELLIDOS
DOMICILIO
NUM. PISO ESCALERA COD. POSTAL
POBLACION PROV.
TELEFONO FIRMA

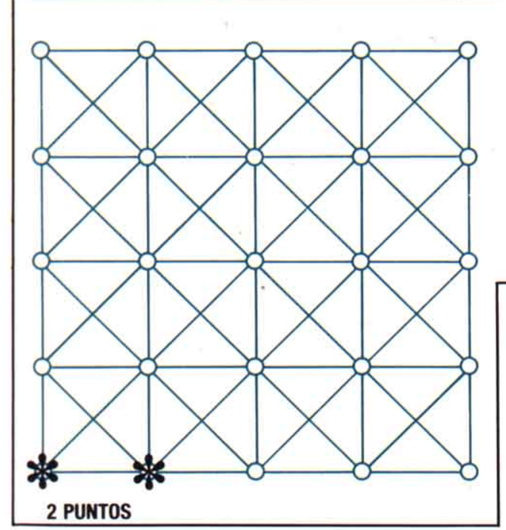
Para que sepa de alguna manera lo que es bueno y lo que es malo, lo que conviene y lo que no conviene hacer, lo que un adversario listo podría hacer, y lo que no haría nunca, es necesario que posea una serie de criterios que le permitan valorar en cifras una situación dada. En el caso del ajedrez, se pueden llegar a considerar hasta sesenta criterios, como por ejemplo los siguientes:

- Libertad de acción (número de jugadas posibles y favorables que podemos hacer)
- Protección de las figuras
- Control del centro del tablero
- Amenaza sobre las figuras enemigas, etc.

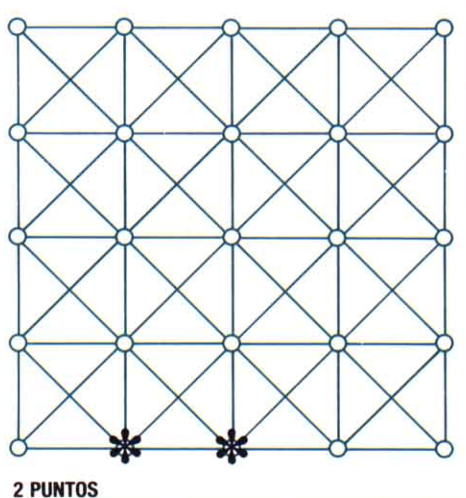
Se hace evidente que los algoritmos capaces de hacer un trabajo tan árduo y minucioso han de ser formidablemente rápidos, ya que de lo contrario el ordenador tardaría una eternidad en darnos su respuesta y el programa perdería así gran parte de su utilidad. Sería un buen ejercicio didáctico ensayar estos conceptos en BASIC, pero los resultados nunca serían satisfactorios. En cambio, en un juego menos complicado, como el «cuatro en raya» o el «tres en raya», la lentitud del BASIC sería un obstáculo perfectamente superable. Además, la sencillez de estos dos juegos nos va a permitir profundizar más en lo que hemos visto hasta ahora.

Como el juego de las tres en raya es quizás demasiado simple, y como además siempre acaba en tablas, vamos a utilizar el de las cuatro en raya, aunque no el tradicional, sino una especie de mezcla entre el primero y el segundo.

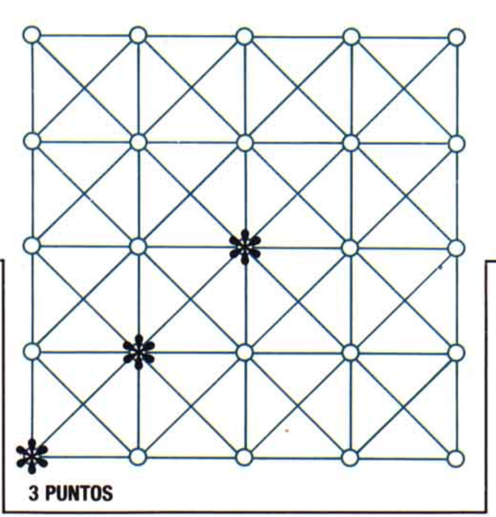
FIGURA 1



2 PUNTOS



2 PUNTOS



3 PUNTOS

En la figura número 1 podemos ver una serie de alineamientos. A cada uno de ellos le otorgamos un valor numerico según sea más o menos bueno, es decir, en razón a las ventajas que presenta con vistas a completar la línea de cuatro cruces:

El primer alineamiento es el más sencillo y también el menos favorable, ya que consta de sólo dos cruces y parte de una esquina. El segundo vale el doble que el primero (dos) porque, al no partir de un lateral, hace posible continuar la línea por cualquiera de sus dos extremos. Por esta misma razón, el cuarto alineamiento es el que más valor tiene, pues basta con colocar otra cruz en uno de los extremos de la línea para ganar la partida (alineamiento 5).

A partir de estos presupuestos, ¿cómo desarrolla la máquina su estrategia?

Antes de responder a la pregunta, conviene que aclaremos un poco el significado de lo que los especialistas llaman «Juegos bipersonales de suma nula». Simplificando, podríamos decir que son aquellos juegos (mejor diríamos «relaciones», pues se trata de una teoría aplicada más a la toma de decisiones sobre situaciones reales, que a los «juegos» (*games*) propiamente dichos), en los que existen dos adversarios, o dos intereses opuestos, de forma que la desventaja en la que se encuentre un jugador determina en qué medida le lleva el otro ventaja y viceversa. Es decir, si puntuamos con números positivos y negativos el valor de la situación es siempre nula. Veámoslo con un ejemplo:

Si el jugador A tiene un alineamiento del tipo 4 (seis puntos), y el jugador B uno del tipo 3 (tres puntos), el valor de la situación de A será

$$6 - 3 = 3$$

y la de B

$$3 - 6 = -3$$

luego

$$3 + (-3) = 0$$

Vayamos ahora a la figura número dos.

El ordenador juega con las blancas y nosotros con las negras. Le toca mover a él. Comienza por ver qué juga-

das puede hacer y cuánto puntúa cada una, llegando a la conclusión de que colocando la ficha en (3,A) obtiene la jugada de valor más alto (tres puntos). Ahora se pone en el lugar del contrario y busca la jugada más favorable que podrían hacer las negras si las blancas colocaran en (3,A). Indudablemente, las negras colocarían en

4

6 PUNTOS

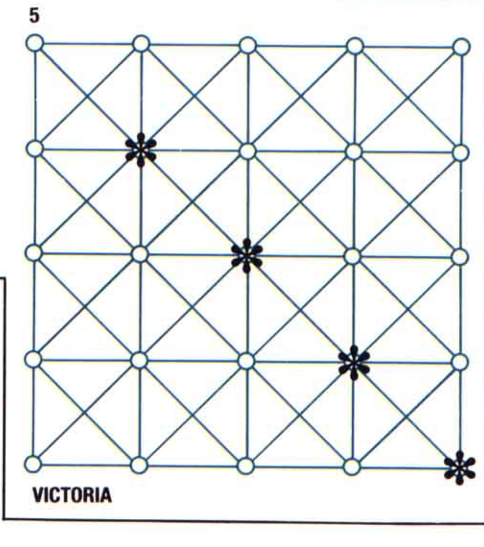
(1,B), obteniendo una jugada de seis puntos. De esta forma, el ordenador sabe que la ganancia del contrario sería mayor que la suya si realizara esa jugada (3 − 6 = −3), y decide, por tanto, «taponar la jugada del contrario» situando un ficha blanca en (1,B). Resumiendo, el ordenador busca su mejor jugada para buscar después la mejor jugada del contrario y restar el valor de ambas; si el resultado es positivo, hace su jugada, y si es negativo, evita que el contrario haga la suya.

Resulta muy evidente lo fácil que sería ganar a un programa que utilizara un algoritmo tan sencillo como éste. Sin embargo, basta decir que ampliando su profundidad de previsión de jugadas futuras y añadiéndole algún criterio de evaluación más, sería prácticamente invencible. Por ejemplo, se podría dar una puntuación especial por la posesión del centro del tablero.

La determinación de criterios de evaluación y la aplicación de algoritmos del tipo «juegos bipersonales de suma nula», son fundamentos válidos para la mayoría de los juegos de estrategia. En esencia, el funcionamiento de los programas de este tipo no difiere en mucho del ejemplo que hemos analizado en este capítulo.

```
10 REM CUATRO EN RAYA.
   ERNESTO DEL VALLE - INPUT.
   1986
20 REM NOTA IMPORTANTE-
   TECLEAR Y EJECUTAR EL
   PROGRAMA EN MODO
   "MAYUSCULAS" (CAPS LOCK).
30 SCREEN 1: TT=1:DIM A(7,4)
40 GOTO 9000
```

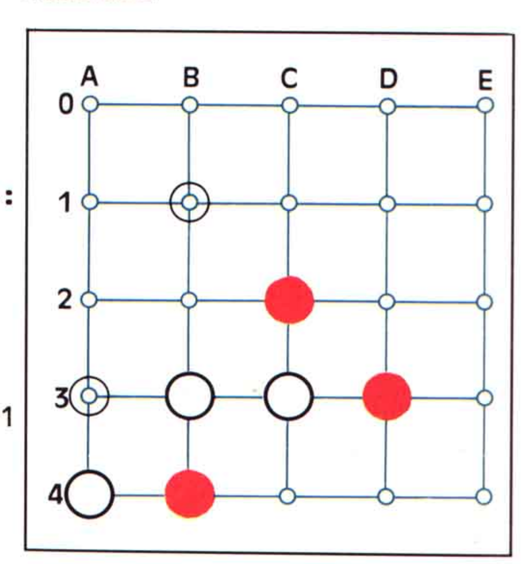
5



VICTORIA

```
1000 REM MI TURNO
1010 IF T=2 GOTO 2000
1020 V=0: C=0: LOCATE 0,1:
     PRINT "MI TURNO      ":
     TF=0: TA=0
1025 IF TT=1 THEN NN=4: FF=4:
     GOSUB 1800: T=2: GOTO
     2000
1028 UU=0
1029 U=0
1030 FOR N=0 TO 7
1040 FOR F=0 TO 4
1050 IF A(N,F)=0 THEN UU=UU+1
     :GOSUB 1100
1060 NEXT F
1061 NEXT N
1062 IF UU=0 GOTO 2500
1065 IF TA=3 AND U=0 THEN V=1
     : GOTO 1800
1068 IF UU=1 GOTO 2500
1070 IF U=0 THEN U=1: GOTO
     1030
1075 GOSUB 1800
1080 TT=TT+1
1090 T=2: GOTO 2000
1100 REM RUTINA DE ANALISIS Y
     VALORACION
1102 IF F=4 GOTO 1105
1103 IF A(N,F+1)=0 THEN
     UU=UU-1: RETURN
1105 C=0: X=N: Y=F
1110 X=X+1: GOSUB 1500
1112 IF R=0 GOTO 1110
1115 X=N: Y=F
1120 X=X-1: GOSUB 1500
1122 IF R=0 GOTO 1120
1125 C=0: X=N: Y=F
1130 X=X-1: Y=Y-1: GOSUB
     1500
1132 IF R=0 GOTO 1130
1135 X=N: Y=F
1140 X=X+1: Y=Y+1: GOSUB
     1500
1142 IF R=0 GOTO 1140
1145 C=0: X=N: Y=F
1150 X=X-1: Y=Y+1: GOSUB
     1500
1152 IF R=0 GOTO 1150
1155 X=N: Y=F
1160 X=X+1: Y=Y-1: GOSUB
     1500
1162 IF R=0 GOTO 1160
1165 C=0: X=N: Y=F
```

FIGURA 2





UNA OPORTUNIDAD IRREPETIBLE

Ahora tienes la ocasión de hacerte con los 10 mayores éxitos del año en su presentación original (cada uno en su estuche y con su carátula) a un precio de auténtico chollo: 3.995 pts.

Imagínate... Los 10 mejores títulos de 1986 por poco más de lo que cuesta uno solo.

¡¡¡PIDE EL SUPER-10 EN TU TIENDA ANTES QUE SE AGOTE!!!

"SUPER-10" SPECTRUM

EXPLODING FIST
TURBO ESPRIT
ROCK'N LUCHA
ZORRO
3 WEEKS IN PARADISE
ABU SIMBEL (PROFANATION)
SABOTEUR
CAULDRON II
BRUCE LEE
SPY HUNTER

"SUPER-10" COMMODORE

EXPLODING FIST
URIDIUM
GOONIES
SABOTEUR
BEACH HEAD II
CRITICAL MASS
SPY HUNTER
ZORRO
SUPER-ZAXXON
FIGHTING WARRIOR

"SUPER-10" AMSTRAD

EXPLODING FIST
TURBO ESPRIT
ROCK'N LUCHA
ZORRO
3 WEEKS IN PARADISE
ABU SIMBEL (PROFANATION)
SABOTEUR
CAULDRON II
BRUCE LEE
RAID OVER MOSCOW

"SUPER-10" M.S.X.

ALIEN-8
KNIGHT LORE
GUNFRIGHT
NIGHTSHADE
JACK THE NIPPER
SHOWJUMPER
VALKYR
BOUNDER
MAPGAME
JET SET WILLY II



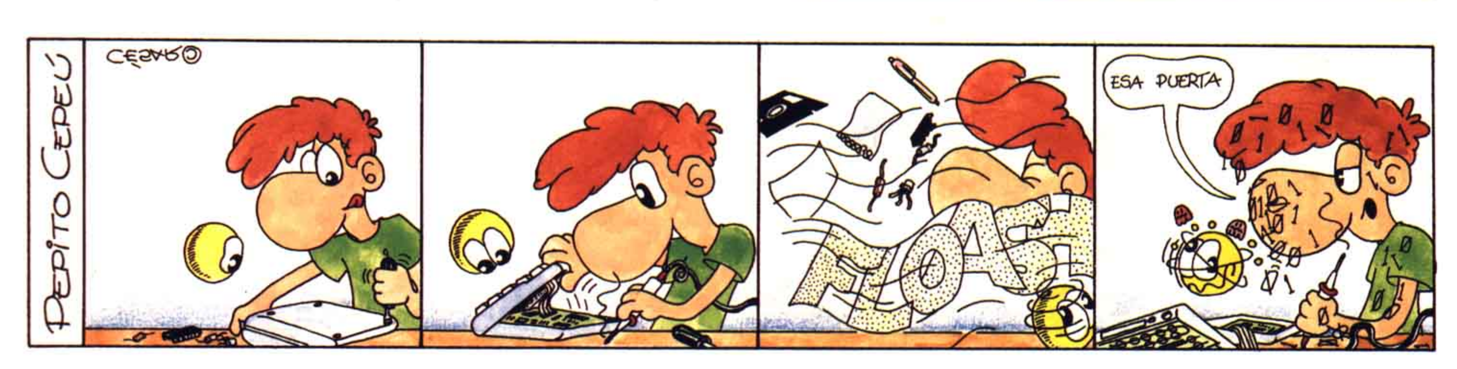
SOLO
3995 PTS.

ERBE Software

DISTRIBUIDOR EXCLUSIVO PARA ESPAÑA: ERBE SOFTWARE
C/. STA. ENGRACIA, 17. 28010 MADRID. TEL. (91) 447 34 10
DELEGACION BARCELONA: AVDA. MISTRAL, 10 TEL. (93) 432 07 31

```
1170 Y=Y-1: GOSUB 1500
1172 IF R=0 GOTO 1170
1175 X=N: Y=F
     R=0 GOTO 1178
1179 C=0: X=N: Y=F
1180 IF TF>TA THEN TA=TF:
     NN=N: FF=F
1190 F=4:RETURN
1500 G=(1 AND X<0)+(1 AND X>7
     )+(1 AND Y<0)+
     (1 AND Y>4)
1502 IF G>0 GOTO 1520
1505 IF A(X,Y)<>1 AND U=0
     GOTO 1520
1508 IF A(X,Y)<>2 AND U=1
     GOTO 1520
1510 R=0: C=C+1
1515 RETURN
1520 IF C>TF THEN TF=C
1530 R=1: RETURN
1800 REM RUTINA FINAL DE MI
     TURNO. COMPROBACION DE
     VICTORIA E IMPRESION DE
     FICHA
1802 IF V<>2 GOTO 1810
1805 CLS: PRINT "ME HAS
     GANADO": PRINT"PULSA UNA
     TECLA": BEEP
1808 IF INKEY$="" GOTO 1808
1809 RUN
1810 X=(NN*3)+2
1815 Y=(FF*3)+4
1820 LOCATE X,Y:PRINT CHR$(1)
     ;CHR$(65)
1830 A(NN,FF)=1: BEEP
1840 IF V<>1 THEN RETURN
1850 CLS: PRINT "TE HE GANADO
     ": PRINT "PULSA UNA
     TECLA": BEEP
1860 IF INKEY$="" GOTO 1860
1870 RUN
```

A continuación publicamos la primera parte de un programa «inteligente», capaz de jugar a las «Cuatro en Raya». En el próximo número lo completaremos con el resto del listado y las instrucciones de manejo.





LOS MEJORES DE INPUT MSX

PUESTO	TITULO	PORCENTAJE
1.º	Knight Lore	17,8 %
2.º	H.E.R.O.	13,4 %
3.º	Soccer	13,4 %
4.º	Profanation	12,3 %
5.º	Alien 8	11,9 %
6.º	Jack the Nipper	8,8 %
7.º	Hyper rally	6,7 %
8.º	Yie ar kung fu	6,3 %
9.º	Road Fighter	5,5 %
10.º	Yie ar Kung fu II	3,9 %
		100 %

Para la confección de esta relación únicamente se han tenido en cuenta las votaciones enviadas por nuestros lectores de acuerdo con la sección «Los Mejores de Input».

Noviembre de 1986.

OTRA CLASE DE SOFT

Desde ahora mismo tienes la posibilidad de trabajar, en tu MSX, **con los programas más famosos de la historia de la informática. Los títulos, versiones en *diskette* de 3.5 pulgadas para el sistema operativo** MSX-DOS, **son tan increíbles como:** Wordstar, dBASE II, Multiplan, Turbopascal,... **¡Es otra clase de *Software*!**

Spectravideo, la primera firma que se ha decidido a comercializar estos excelentes programas, en versión **MSX**, nos ha hecho llegar unos cuantos de los nuevos títulos que desde ahora mismo pone a disposición de todos los usuarios del estándar.

Todos los programas vienen en versión *diskette* de 3.5 pulgadas y preparados para trabajar bajo el sistema operativo de disco **MSX-DOS**. Es probable que nunca lleguemos a ver versiones en *cassette*, puesto que se trata de programas que requieren frecuentes accesos al *diskette* para leer y escribir información (o partes determinadas del propio programa, que van cargándose en memoria de forma sucesiva, sólo cuando se hace necesario, en lo que se conoce como técnica de *overlays*). El uso de un *cassette* haría prácticamente imposible el funcionamiento de este excelente *soft* y, aunque fuera posible conseguirlo, sería a costa de la velocidad y conduciría a versiones excesivamente lentas.

Los programas funcionan perfectamente tanto en las máquinas de la primera como en las de la segunda generación **MSX** (lo único necesario es la citada unidad de *diskettes*). Probablemente, el mayor problema para los usuarios de máquinas de la primera generación sea el no disponer del formato de 80 columnas.

Todos los programas están pensados para trabajar en este formato y aunque pueden hacerlo perfectamente en 40 columnas, es a costa de la calidad en la presentación.

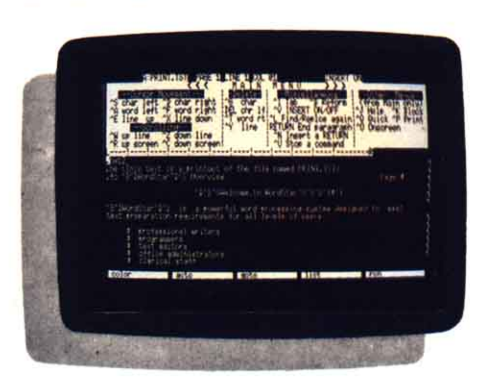
En cualquier caso, los usuarios de la primera generación pueden hacer uso de alguno de los cartuchos de 80 columnas disponibles en el mercado y sacar con ello el máximo partido de cada una de estas obras maestras del *soft*.

En artículos sucesivos iremos comentando a fondo cada uno de estos programas, haremos pruebas de compatibilidad y trataremos aspectos particulares de cada versión. Pero no queríamos dejar pasar la oportunidad de daros una panorámica completa de este nuevo *soft* (nuevo en su versión **MSX**), que sin duda abrirá nuevos horizontes a muchos usuarios con inquietudes.

WORDSTAR

Para quien no conozca este nombre legendario, diremos que se trata del programa de tratamiento de textos más famoso de la historia de la informática. Para los usuarios de **MSX** es una novedad que, sin duda, desplazará a todos los programas de tratamiento de textos comercializados hasta ahora.

La versión para **MSX-DOS** es perfecta. Cumple con todas las características del programa, es veloz, completa, viene bien documentada, incluye todas las opciones de las versiones para máquinas de 16 bits y convierte al **MSX** en una poderosísima herramienta para la edición, correción, almacenamiento e impresión de todo tipo de textos escritos, documentos, cartas, etc.



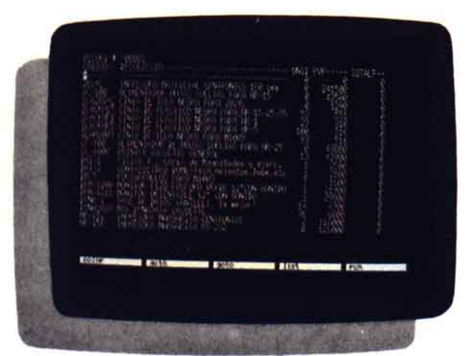
En este caso, el uso de las 80 columnas, aunque no imprescindible, se hace verdaderamente necesario si lo que se desea es trabajar con comodidad.

Otro aspecto interesante es la compatibilidad que existe entre los ficheros de textos almacenados por esta versión y los de los ordenadores **PC** (compatibles **IBM**). En las pruebas realizadas y con la ayuda de una unidad de *diskettes* de 5.25 pulgadas pudimos transferir textos entre **PCs** y **MSX** sin que surgiera la menor dificultad.

En definitiva, un programa de tratamiento de textos con características altamente profesioneles y con unas posibilidades que dejan fuera de juego a los programs de este tipo existentes en la actualidad.

dBASE II

Se trata de uno de los mejores programas de gestión de bases de datos desarrollado para ordenadores personales. Basado en los modelos relacionales y ajustándose bastante bien a lo que se conoce como un modelo completo, **dBASE II** ofrece un entorno para trabajar con varias bases de datos, cada una de ellas formada por un conjunto de ficheros relacionados, en los que podremos guardar toda la información que deseemos a través de registros y campos.



De nuevo, la versión **MSX** es perfecta. Se ha sacrificado algo en cuanto a los límites máximos de utilización (número máximo de registros, etc.) pero nada o casi nada en presentación, velocidad y capacidad de gestión.

De nuevo hay que hablar de compatibilidad con los **PCs** a nivel de ficheros ASCII.

Como comentario basta decir que esta versión deja obsoletos a todos los programas de gestión de bases de datos que existen para **MSX**.

MULTIPLAN

Junto a un programa de tratamiento de textos y un sistema de gestión de bases de datos, no podía faltar un buen programa de los denominados «hoja electrónica» u «hoja de cálculo».

En este caso **Spectravideo** nos ofrece la versión de **Multiplan**, uno de los programas de mayor éxito en máquinas de 16 bits y de los que mayor po-



pularidad han alcanzado entre toda una generación de ejecutivos, empresarios, profesionales, etc.

Aunque se ha sacrificado algo de capacidad, por las limitaciones de memoria de los **MSX**, el resto de las características del programa siguen intactas.

TURBOPASCAL

Este programa, de la firma **Borland**, está reputado como la mejor versión de lenguaje PASCAL que se hay diseñado nunca para ordenadores personales.

El programa proporciona un excelente entorno para la programación en este lenguaje. Entre los programas incluidos hay que destacar un editor (que ha llegado a cobrar fama por sus posibilidades como programa de tratamiento de textos) y un velocísimo compilador, que en cuestión de segundos compilará los programas fuente, en lenguaje PASCAL, introducidos desde el editor.

Para hacerse una idea de las posibilidades de este increíble programa, podemos mencionar que un gran número de serios proyectos de progra-



mación, tanto en la empresa privada como en la universidad, se han desarrollado y se siguen desarrollando con esta versión del lenguaje PASCAL, eso si, utilizando máquinas con CPUs de al menos 16 bits.

LOS PROGRAMAS NEVADA

Se trata de un conjunto de cuatro programas, tres de ellos dedicados a otros tantos lenguajes de programación y el cuarto destinado a la edición de textos, de la firma estadounidense **Ellis Computing**. Sus nombres son: **Nevada FORTRAN**, **Nevada COBOL**, **Nevada PASCAL** y **Nevada EDIT**.

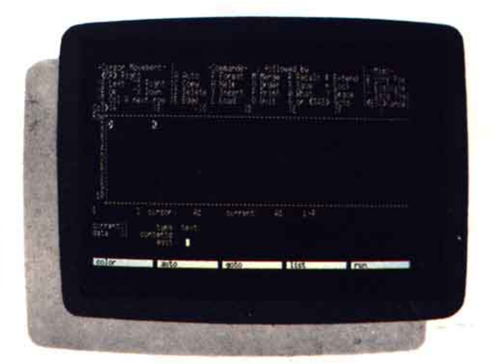
Los tres primeros están dedicados a los lenguajes de programación que les dan nombre, e incluyen una serie de subprogramas que dan origen a los distintos entornos de programación. En todos ellos hay un editor para la confección de los programas fuente y un compilador que se encarga de la transformación del programa fuente en programa objeto.

Son programas de altas prestaciones, con una presentación verdaderamente profesional y unas excelentes características de velocidad, flexibilidad, sencillez de manejo, etc. En este primer acercamiento diremos sobre todo que se trata de versiones muy superiores a las que circulan actualmente por el mercado, comparables en calidad a la mencionada versión de **Turbopascal**.

Nevada EDIT, por su parte, es un sencillo editor de textos para **MSX-DOS**, destinado a servir como editor de propósito general, tanto para la edición de programas fuente, de ficheros de comandos e incluso de ficheros de textos sencillos.

ALGO MAS DE MICROPRO

De **Micropro**, los creadores de la versión **Wordstar** arriba mencionada, **Spectravideo** comercializa además los programas **Calcstar**, **Datastar**, **Reportstar**. Se trata respectivamente de una hoja electrónica, un sistema de





gestión de bases de datos y un programa para la generación de informes.

Las primeras impresiones sobre estos programas son que mantienen la calidad y las excelentes prestaciones de la versión **Wordstar**, muy superior a los niveles a que nos tienen acostumbrados en **MSX**.

Habrá más comentarios y un análisis en profundidad de este soberbio conjunto de programas.

EL CABALLERO DE LA TABLA POLIGONAL

Con este juego te transformarás en el **Caballero Mágico**. Tu deseo de aventuras te ha llevado hasta el reino de **Ibisima**.
El rey de **Ibisima** no sabe qué regalarle a su hija **Germinitrude**. Tu misión será encontrar un regalo digno de ella en el castillo de **Duendilandia**. Sólo así lograrás el favor del rey que, en agradecimiento, te nombrará miembro de la **Tabla Poligonal** de **Ibisima**.
Nada más comenzar serás transportado al castillo, donde tendrás que arreglártelas tú solo, buscando el camino adecuado a través de todas sus salas.
En ellas encontrarás monstruos terribles y criaturas que continuamente se cruzarán contigo robándote tu energía. ¡Cuidado, pués si el **Caballero Mágico** pierde su energía, morirá!
En todas las salas irás encontrando objetos, que podrás examinar y recoger si te interesa, para más tarde poder comerciar con los mercaderes fantasmas. Ellos te ofrecerán dinero por tus tesoros, y tu podrás vendérselos si lo deseas.
¡Cuidado con los laberintos! La reina los ha llenado de trampas para que no puedas llevar el regalo a la hija del rey.
Al final, podrás elegir entre dos opciones: o escaparte del castillo de **Duendilandia** con todas las riquezas que puedas, o entregárselas al rey, para que éste se las de a su hija. De ti depende escoger la acción más adecuada... ¡Suerte, futuro caballero de la **Tabla Poligonal**! Te auguramos ratos muy divertidos en el reino de **Ibisima**.

DATOS GENERALES

TITULO Finders Keepers
FABRICANTE Mastertronic
CLASE DE PROGRAMA Juego
FORMATO Cassette

CALIFICACION (Sobre 10 ptos.)	
ORIGINALIDAD	8
INTERES	8
GRAFICOS	8
COLOR	8
SONIDO	7
TOTAL	39







****

LAS TRES LUCES DE GLAURUNG

Se trata de un estupendo programa, adaptado a **MSX** desde su versión original para el ordenador **Spectrum**, que narra las aventuras de un intrépido caballero, perdido en alguna de las vastas y silenciosas salas del tenebroso dominio del señor del mal...
Este argumento hace referencia a la primera característica del juego, que es uno más de la interminable serie de programas en los que un personaje se mueve explorando las salas de alguna construcción, más o menos extraña y compleja.
En su exploración a través de las distintas habitaciones, el jugador tiene que encontrar tres joyas y una llave. Con estos elementos en la mano y sólo cuando tenga todos ellos, el jugador podrá acercarse hasta la salida y escapar.
Para entorpecer esta búsqueda y evitar que el protagonista escape fácilmente, los programadores han incluido una serie de enemigos, de distintas características, a los que

tendrá que enfrentarse el jugador, ya sea directamente con sus armas o utilizando alguno de los objetos que consiga recoger.
El arma más poderosa con que cuenta el jugador es su arco. Las flechas que dispare con él neutralizarán a la mayoría de los enemigos, salvo a dos de ellos: **el Mago** y **el Dragón**. Para derrotar a estos es necesario además llevar alguna de las tres joyas.
Estas joyas y el resto de los objetos que el aventurero puede transportar, los encontrará en el interior de una serie de cofres desperdigados por las habitaciones. Para dar más emoción al asunto de los cofres, algunos de ellos, en lugar de contener un objeto útil, están llenos de desagradables sorpresas, así que cuidado con ellos.







DATOS GENERALES

TITULO Las tres luces de Glaurung

FABRICANTE Erbe

CLASE DE PROGRAMA Aventura en el castillo

FORMATO Cassette

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	7
INTERES	7
GRAFICOS	7
COLOR	8
SONIDO	5
TOTAL	34





El apartado gráficos merece algunos comentarios. Al ser una adaptación de la versión original **Spectrum**, hay bastantes aspectos de este ordenador, que podrían haberse resuelto de forma mucho más brillante aprovechando las posibilidades de los **MSX**.
Los gráficos y los escenarios son bastante buenos, y hacen buen uso del color. Quizás lo más flojo sea el movimiento de los distintos personajes, que resulta poco suave y un tanto incontrolable, sobre todo cuando se desea saltar.
En conjunto resulta un programa bastante bueno, de características medias, pero sin pegas. Todos los aspectos conflictivos han sido resueltos a la perfección. El número y el aspecto de las distintas salas que componen el mapa es un punto fuerte que por sí solo proporciona la necesaria variedad.
Otro aspecto interesante es que se puede avanzar y pasar de una sala a otra sin excesiva dificultad, al menos en las primeras salas.

****

UNO, DOS, ARRIBA, ABAJO

En los tiempos que corren, tener unos kilos de más o simplemente no estar en forma, significa estar pasado de moda. **Idealogic** lo sabe y por ello, uno de los primeros títulos de su línea de programas para **MSX2**, creado especialmente para el segunda generación de **Philips**, es este divertido **Aerobics**, un programa original con el que ambas empresas intentan ayudarnos a mantenernos en forma.
Aerobics nos presenta en la pantalla del televisor a una simpática «profesora», que se encarga de mostrarnos cómo se hacen los distintos ejercicios, de acompañarnos en su ejecución y de dirigir, en fin, toda la sesión de *aerobic*.
El programa, del que nos ha llegado una primera versión en *diskette* de 3.5 pulgadas, comienza, tras una presentación con imagen digitalizada de los autores, por ofrecernos un menú con tres niveles de dificultad de los ejercicios.
Seleccionado uno de ellos, da

comienzo la sesión con la aparición de la profesora sobre un fondo que representa a un gimnasio, al tiempo que empieza a sonar alguna de las



melodías disco incluidas en el programa y destinadas a marcar el ritmo de los ejercicios.
La sesión comienza por un precalentamiento y sigue con una



serie de ejercicios específicos destinados a las distintas partes de nuestra anatomía (brazos, piernas, cintura, estomago, nalgas,...), y así hasta que termina la sesión.



Seguro que alguno está pensando que lo que hay que hacer es sentarse en la poltrona, coger el *joystick* e intentar los ejercicios a base de muñeca. Pues no, de lo que se trata es de, después de habernos puesto el chandal, imitar los movimientos que la profesora hace para nosotros. Gráficos, animación y sonido son las bases de todo el programa, uno de los primeros que intenta aprovechar a fondo las posibilidades de un **MSX2**. El modo gráfico elegido ha sido SCREEN 7 (512×212 puntos en 16 colores) y el mayor logro es la animación de la imagen móvil de la profesora. Al parecer, el equipo de diseño gráfico se dedicó a estudiar

DATOS GENERALES

TITULO Aerobics

FABRICANTE Idealogic

CLASE DE PROGRAMA Para estar en forma

FORMATO Cassette o Diskette

CALIFICACION (Sobre 10 ptos.)

ORIGINALIDAD	9
INTERES	10
GRAFICOS	10
COLOR	9
SONIDO	10
TOTAL	48

las distintas posiciones de cada uno de los ejercicios, con el fin de reproducir lo más fielmente posible los movimientos reales. Algo flojo resulta el escenario de fondo, único e inmóvil, que acaba resultando demasiado conocido. El programa habría ganado mucho en variedad de haber incluido algún escenario más. La música disco que acompaña los ejercicios, ha sido compuesta por un profesional, perteneciente al conjunto español de nombre **Atlanta**.

Los distintos temas, que sincronizan perfectamente con el movimiento de la profesora incluso al variar la velocidad del mismo, están



magníficamente realizados aunque, (quizás sea inevitable por las limitaciones del *chip* de sonido) resultan en conjunto poco brillantes.



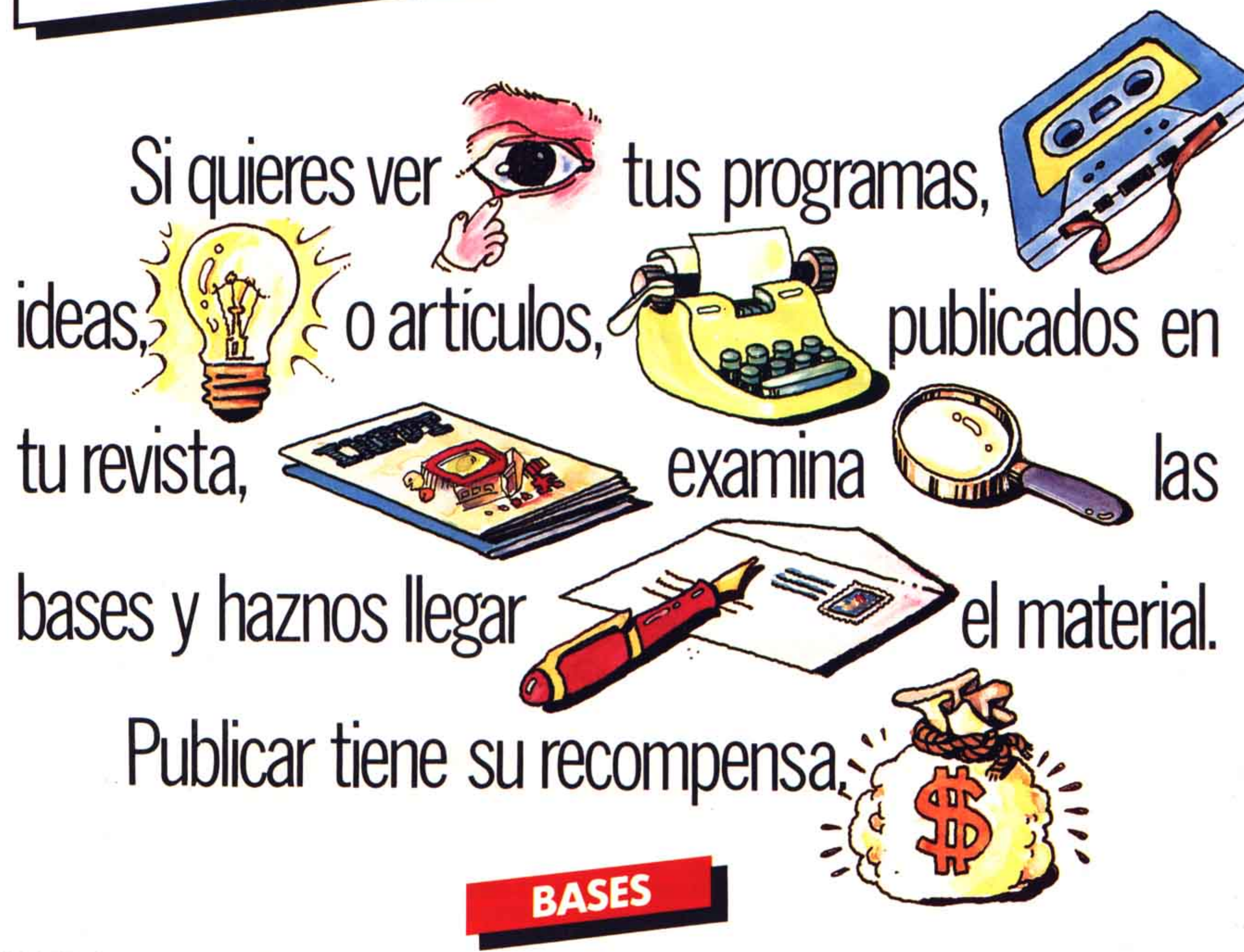
Los distintos niveles de dificultad, la secuencia y la duración de los diferentes ejercicios, temas que supervisó una conocida revista dedicada a la gente sana, son



también algunos de los puntos fuertes de este *soft*.

★★★★★★★★★★★★★★★★★★★★★★★

¡PARTICIPA EN INPUT!



BASES

PROGRAMAS: Una vez desarrollado tu programa, que debe ser original y no haber sido enviado a ninguna otra publicación, puedes enviárnoslo aquí grabado en cassette, diskette o microdrive. Es preferible que vaya acompañado por un listado de impresora, pero no es imprescindible.

El programa habrá de venir acompañado por un texto que aclare cuál es su objetivo, el modo de funcionamiento y una explicación del cometido que cumplen las distintas rutinas que lo componen. El texto se presentará en papel de tamaño folio y mecanografiado a dos espacios. No importa que la redacción no sea muy clara y cuidada; nuestro equipo de expertos se encargará de proporcionarle la forma más atractiva posible.

ARTICULOS E IDEAS: Se aplica lo anteriormente dicho para los textos que acompañan a los programas; es decir, conviene detallar al máximo lo que desees que aparezca publicado en la revista, de la manera que te gustaría que otra persona hubiera explicado eso mismo.

UN JURADO propio decidirá en cada momento qué colaboraciones reúnen los requisitos adecuados para su publicación, y evaluará la cuantía del premio en metálico al que se hagan acreedoras.

No olvidéis indicar claramente para qué ordenador está preparado el material, así como vuestro nombre y dirección y, cuando sea posible, un teléfono de contacto. Entre todos los trabajos recibidos durante cada mes SORTEAREMOS:

- **Un premio de 50.000 ptas.**
- **Un premio de 25.000 ptas.**
- **Un premio de 10.000 ptas.**
  en material microinformático a elegir por los afortunados.

¡No os desaniméis!, por muy simples o complejas que puedan parecer vuestras ideas, todas serán revisadas con el máximo interés.

INPUT MSX
Paseo de la Castellana, 93. Planta 14
28046 MADRID

NOTA: INPUT no se responsabiliza de la devolución del material que no vaya acompañado por un sobre adecuado con el franqueo correspondiente.



EL ZOCO

Vendo ordenador Spectravídeo SVI 728 MSX + manual inglés + manual español + libro de programación + cartucho Track & Field I (Sony) + 40 programas de los mejores por 40.000-45.000 ptas. a convenir. También cambio-vendo juegos en cinta, poseo más de 80 títulos. Llamar o escribir a:

Jordi Ferran
Matajudáica, 10
Tel. (972) 63 01 70
De 8 a 22 horas
Girona

Cambio Pitfall II por cualquier otra cinta de Activision o Zaxxon.

Daniel Manero
Tel. (91) 797 48 21

Vendo/Cambio programas MSX, primeros títulos del mercado.

Iñaki Fernández Izquierdo
Zamakoa, 7, 5° dcha.
Tel. (94) 456 33 72
Galdácano (Vizcaya)

Intercambio y vendo programas, tanto de juegos como de aplicaciones: Alien 8, Abu Simbel, Zen Assembler, Base de Datos, etc.

Suintila García Pulido
San Marcos, 7, 9-5
Badalona (Barcelona)

Vendo por 1.000 ptas. más gastos de envío juegos como: Stop the Express, Pastfinders, Sorcery, Le Mans, etc. o cambio por Zaxxon, Nightshade, Gunfright o Knight Lore. Llamar o escribir a:

José Ramón Carrillo
Montornés 33-35, 1°, 1ª
Tel. (93) 211 21 97
08023 Barcelona

Intercambio juegos (más de 100). Todos primeras marcas (Hiper Sport 1,2,3, Yie Ar Kung-Fu 1 y 2, Roller Ball).

Ricardo Pérez
Tel. (965) 80 02 82

Cambio juego Le Mans (carreras, muy interesante, gráficos 3D) 48K cassette, por The way of the Tiger, Turmoil, Pit fall II, Knight Lore, Hero o alguno de karate. Además vendo libro MSX-Logo por el módico precio de 700 ptas. (135 páginas). Llamar a:

Juan Ramón Oriach
Tel. (973) 74 03 16 (de 9.15 a 10.30)
Almacellas (Lérida)

Vendo SVI 328 nuevo, con Datacassette, joystick, más de 30 programas como: Sasa, Ninja, etc. y revistas, monitor f. verde Hantarex Boxer 12 pulgadas y amplificador de volumen por 75.000 ptas.

Javier
Provenza, 433, 2°, 1ª
Tel. 318 76 20
08025 Barcelona

¡Oferta! Vendo ordenador Philips VG8010. Impecable, sin usar. Manuales y garantía originales. Además regalo listados de juegos, Revistas MSX, y 50 juegos. Todo por 29.980 .

Alberto Frías Pardo
Julián Gayarre, 2
Castejón (Navarra)

Cambio o compro juegos. Escribir a:

Alejandro Gregorio Panizo
Belianes, 20
28043 Madrid

Intercambio programas en código máquina para MSX y C-64. Buenos títulos.

Rafael Sabariego
Plaza Uzturre, 6, 8°C
Tel. (943) 36 38 34
Lasarte (Guipuzcoa)

Intercambio juegos chicos/as. Tengo primeros títulos: Knight Lore, Alien 8, Pit Fall II, etc. Si os interesa escribir a:

Laureano Ros
Navaro de Haro
Tel. 57 76 11
Torre Pacheco (Murcia)

Contacto con chicos-as de toda España. Poseo más de 180 juegos (Profanation, Tiger, etc.) Escribir a:

Luis A. de la Fuente
Gran Capitán, 3-5, 2° dcha.
Salamanca

Intercambio programas MSX, poseo primeros títulos del mercado. Escribir o llamar a:

José Luis de la Pedraja
José Mª Pereda, 49, 1°
Tel. (942) 88 17 61
Torrelavega (Cantabria)

Vendo Data-Cartridge sin estrenar. Precio a convenir, también vendo juegos. Primeras marcas del mercado de MSX.

Armando Palet Martínez
Tel. 676 12 44
Torrejón de Ardoz
Madrid

Vendo grabador Sony TCM-2 nuevo con cable ordenador y un juego por 7.000 ptas. contra reembolso. También intercambio programas de todo tipo para MSX.

José Borrego
Martí L'Huma, 7
Olot
17800 Gerona

Vendo 9 juegos primeros títulos por 3.500 ptas. Los títulos son: Knight Lore, A.R.M., Harrier Attack, Mutanta Monty, Manic Miner, Roload, Comando II, Quack a Jack y Roland Ahoy. Escribir a:

Ignacio Aguilera Espejo
Sevilla, 16
Mairena de Aljarafe
Sevilla

Intercambio programas, últimos títulos. Busco preferentemente: Taewondo, The way of the Tiger, Circus Charly, Hyper Sports III y Mopiranger.

Oscar Frades
Avda. Arqueros, 49, 4°A
28024 Madrid

Vendo enciclopedia de la Informática Basic (Editorial Forum) 6 tomos Basic encuadernados más 3 tomos Basic Advanced (Software para Apple, Commodore y MSX) Sesenta por ciento su valor real. Impecable estado.

Jesús Blanch
Central Telex Telégrafos
Pl. País Valenciano
Tel. (96) 363 34 18 (4 tarde)
46002 Valencia

Cambio programas MSX. Tengo Knight Lore y Ghostbuster. Me gustaría conseguir Hero, The Way of the Tiger,... Escribir a:

Pedro Ramírez Estrella
Industria, 14, pta. 1
Tel. (96) 323 40 81
46022 Valencia

Intercambio programas MSX. Poseo más de 200 títulos en código máquina. Escribir a:

Emilio Rabásco Jiménez
Murcia, 1
Tel. (957) 25 03 41
14010 Córdoba

Intercambio todo tipo de programas sin interés económico. Poseo unos 70 (sólo Barcelona ciudad).

Francisco José García
Bonavista, 31-33 4° 1ª
08012 Barcelona

EL ZOCO

Vendo el siguiente material: 3 cartuchos (Socer, Hesware, Simon's Basic) y 200 juegos tales como Rambo y Comando y utilidades tales como Simulador de Spectrum Busicalc, etc. También vendo las siguientes revistas: Nuevas tecnologías número 1, Biblioteca de Informática nº 1, Lenguaje Máquina del Commodore 64, Muy ordenadores nº 4,5 y 6, Commodore World nº 17,19,20 al 26 y el 28; Commodore Magazine nº 12,23 al 28; Basic nº 121; Input Commodore del 1 al 6 y el 8; Mi computer nº 21,23,25; El ordenador personal nº 24,22,41. Los juegos y cartuchos son del C-64. Todo aquel que tenga interés en los juegos, cartuchos, revistas que dirija sus cartas a:

Miguel Angel Huelves García
Entre Arroyos, 64, Ib
Tel. 439 96 31
28030 Madrid

Vendo programas MSX. Tengo Knight Lore y Ghostbuster. Los vendo a 500 ptas. cada uno. Dirigirse a:

Pedro Angel Serrano Soguero
Avda. Puerto 234, pta. 21ª
46023 Valencia

Desearía contactar con usarios de MSX de Jaen, para intercambiar ideas, programas (tengo Chuckie Egg, Grand National, Nick Neader, Champ (Ensamblador), y varias cintas de utilidades. Interesados dirigirse a:

Pedro Peraba Montiel
Jimenez Serrano, 2
Tel. (953) 26 56 12
Jaen

Vendo cartucho de ampliación de memoria HBM-16 de Sony por valor de 4.000 ptas.

Luis Santiago Moleón Garrido
Trajano, 5, 5ºF
Tel. 20 57 77
18002 Granada

Desearía cambiar los programas Zaxxon y Jet Set Willy II por H.E.R.O. o Abu Simbel, Profanation.

Sergi Bosch
Tel. (93) 379 20 86

Cambio y vendo programas MSX, poseo todos los cartuchos de Konami y muchos cassettes.

Javier Galán
Tel. 332 54 29
Barcelona

Desearía intercambiar todo tipo de juegos para MSX. Interesados llamar a:

Luis Crespo
Avda. Mediterraneo, 28, 1ºA
Tel. (93) 718 59 99
Ciudad Badia (Barcelona)

Cambio cartuchos para video-juegos Phiips (Comecocos, Fórmula 1, Baloncesto, Bolos, Marcianitos y Ajedrez, con su correspondiente memoria para jugar contra el aparato) por un monitor o televisor portátil.

Antonio Veas González
Tel. 330 09 17
Barcelona

Vendo ordenador Yashica YC-64 con 64K de RAM y 16K VRAM. Comprado en Marzo de este año, por 30.000 ptas. Interesados llamar a:

Félix
Tel. (985) 21 79 45
Oviedo

Vendo ordenador Hit Bit Sony con cartucho de ampliación 16K, por 30.000 ptas. Para más información:

José Manuel Trías Company
Barne, 3
Tel. 63 18 33
Sóller (Mallorca)

Vendo ordendor HB-55 P de Sony con ampliación de 16K por 23.000 ptas. o con ampliación de 64K por 33.000 ptas. Doy cintas L'n'RUN y 15 revistas MSX. Llamar a:

Angel Ois
Tel. (981) 21 35 53
La Coruña

Intercambio programas MSX, Disc Warrior, Alien 8, Jet Fighter por Jack the Nipper, Bc. Grog Revenge II o The Dambusters. Llamar o escribir a:

José Vicente Mas
Mateo Fial, 31
Tel. (971) 23 26 77
Palma de Mallorca
07013 Baleares

Intercambio programas MSX. Poseo H.E.R.O, Ninja, Scentipede, Jack the Nipper, Profanation, etc. Interesados escribir a:

Rafael Manresa
Buenos Aires, 6, 2 dcha.
30011 Murcia

Vendo lapiz óptico 3.000 ptas., Interface MK2 1.000 ptas., Interface con sonido 1.500 ptas., y muchísimos juegos originales a 995 ptas. cada uno. Todo para Spectrum.

Alfonso Lucas Gómez
Paseo Julio Urquijo, 32, 7ºA
Tel. 39 86 87
Bidebieta
20016 San Sebastián

Vendo o cambio copión para SVI-728 por 1000 ptas. o por uno de estos juegos para MSX: Camelot Warriors, Green Beret, The Way of the Tiger, Jack the Nipper, Rambo o Sobre Wulf.

F. Artal Barba
Avda. Madrid, 32 7ºA
20011 San Sebastián
Guipuzcoa

Intercambio programas en MSX. Entre otros: Time Pilot, Fitching Rider, Hiper 3, Basketball, Baseball, etc.

Guillem Carreras García
Via Augusta, 320-322, 2º 3ª
Tel. 203 54 30
08017 Barcelona

Vendo ordenador Sony HB101P-48Kb. (comprado en enero) más ampliación Sony de 16Kb.. Todo en garantía por 40.000 ptas.

Diego D. González Rae
Avda. Zamora 31, 2º izda.
Tel. (986) 41 73 01
Vigo (Pontevedra)

Vendo o cambio juegos y programas del MSX. También compro unidad de diskettes de 3 1/2 pulgadas, precio a convenir. Escribir o llamar a:

José Antonio Rodríguez
Avda. America 66
Tel. 246 61 79
Madrid

Cambio juegos de MSX. Tengo Zaxxon, H.E.R.O., Decathlon, etc. Escribir a:

Eduardo Laplaza
Junceda, 6 5 I
08101 Rubi

Intercambio y vendo ideas, trucos y programas para ordenadores MSX.

Pedro José Ruiz Yelo
Pza. Constitución, 1, 1ºD edif. Alianza
30820 Alcantarilla (Murcia)

Busco ordenador MSX, unidad de disco, grabador cassette, joystick y demás periféricos, programas en cassette o disco y cualquier consejo. Soy completamente novato. Espero ofertas.

Armando
Apartado 274
Talavera
45600 Toledo

Desearía contactar con usuarios del MSX, para intercambiar opiniones, trucos, gráficos, sonido, preferiblemente de Vigo.

Francisco J. Fernández Martínez
Tel.(986) 41 77 07
Vigo (Pontevedra)

Cambio juegos MSX de los mejores: H.E.R.O., Gunfright, Yie Ar Kung Fu, River Raid, etc.

Miguel A. García Martínez
Carrasco, 2, 7º izda.
Tel. (965) 85 47 57
Benidorm
03500 Alicante



MSX LENGUAJE MAQUINA

Autor: H. Dullin y H. Strasenburg
Editor: Ferre Moret S.A.
Paginas: 305
Precio:

Muchas veces habremos sentido la necesidad de dar a nuestros programas una rapidez que sólo con el BASIC no es posible conseguir. Para ello es necesario programar en lenguaje máquina. Sin embargo, hay que disponer de una información adecuada del microprocesador **Z80A**, de sus instrucciones en código máquina y de su lenguaje ensamblador.

Asimismo es conveniente tener herramientas básicas para poder desarrollar programas, como son un programa ensamblador, un desensamblador, y un simulador del comportamiento del programa, para poder seguir su ejecución y encontrar posibles errores.

Todo esto que acabamos de mencioanr se encuentra incluido en el libro **MSX Lenguaje Máquina**. En él se nos ofrecen en primer lugar unos conceptos generales que comprenden las ideas de lenguaje máquina y de algunos sistemas de numeración: decimal, binario y hexadecimal.

A continuación se dedican tres capítulos al microprocesador **Z80A**, corazón de los ordenadores **MSX**, explicándose su estructura interna y su juego de instrucciones.

Se dedica a continuación un capítulo a técnicas de programación en lenguaje máquina y otro, el último, a perspectivas futuras.

El epílogo contiene útiles tablas de conversión y tablas de comandos.

El tratamiento del tema es completo y didáctico. Los conceptos se explican de una forma muy clara y sencilla de forma que el lector, aunque carezca de conocimientos previos, puede aprender mucho y rápido.

Al final de cada capítulo se incluyen ejercicios sencillos para que el lector pueda comprobar la asimilación de conceptos.

Hay que destacar tres útiles programas cuyos listados aparecen también en este libro. Se trata de un ensamblador, con el cual evitaremos tener que cargar los programas desde el BASIC utilizando los códigos hexadecimales (el tedioso procedimiento de los DATA), pudiendo usar los términos del ensamblador del **Z80**, más fáciles de manejar; hay también un desensamblador que nos permitirá ver los mnemónicos del programa escrito en memoria y un simulador paso a paso, con el cual podremos seguir la ejecución del programa instrucción por instrucción y descubrir así posibles errores.

LENGUAJE MAQUINA PARA MSX, INTRODUCCION Y CONCEPTOS AVANZADOS

Autor: Joe Pritchard
Editor: Anaya
Páginas: 240
Precio:

Dentro de la serie de libros dedicada por **Anaya** al sistema **MSX**, el que presentamos está dedicado a la programación en lenguaje máquina.

Es una obra pensada para el usuario de **MSX** que no posee conocimientos previos de esta técnica de programación, y por ello, el nivel de las ideas explicadas es bastante básico y la forma de exponerlas muy sencilla, haciendo frecuente uso de ejemplos.

El libro empieza por desarrollar una serie de principios básicos, con una introducción al **MSX**, y a la forma de calcular de los ordenadores, centrándose luego en el microprocesador **Z80**, sus registros, modos de direccionamiento, instrucciones de transferencia, de control y de entrada-salida. A lo largo de estos capítulos se incluyen frecuentemente ejemplos, con los que probar y comprobar de manera práctica los resultados del último concepto explicado.

Hay a continuación dos capítulos de indudable interés. Uno de ellos está dedicado al procesador de video (VDP), y otro al generador programable de sonido (PSG). Es muy importante la inclusión de estos capítulos en cualquier libro que trate de programación en lenguaje máquina, ya que las necesidades fundamentales para el uso de esta forma de programación surgen de los gráficos y del sonido que deben ser gestionados rápidamente (sobre todo en juegos), y en ello el lenguaje máquina no tiene rival.

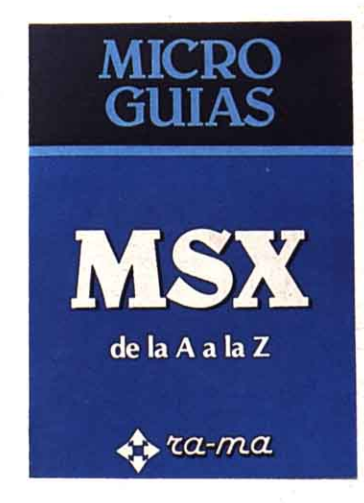
El principal defecto que puede achacarse a este libro es la excesiva simplicidad en las ideas expuestas y en la forma de exponerlas. Ello puede hacer que el libro se quede corto para algún lector.



MSX DE LA A A LA Z

Autor: Varios
Editor: RA-MA
Páginas: 233
Precio:

Se presenta en este libro un compendio ordenado alfabéticamente de todas las palabras y términos que tienen relación con el **MSX**.



En el podemos buscar conceptos generales, tales como notación hexadecimal, intérprete, interrupción, etc., que nos ayudarán a clarificar nuestras ideas básicas. Están también incluidas las palabras del **MSX-BASIC**, acompañadas, cuando esto es posible, de un pequeño programa de ejemplo, para ilustrar su uso.

Asimismo, en caso de que decidamos pasar a programar el microprocesador, **Z80A** directamente en código máquina, tenemos recopilada la definición de cada uno de los mnemónicos del lenguaje ensamblador.

Por último, aparecen también conceptos del sistema operativo en disco **MSX-DOS**, con esto se abarca de una manera bastante global la totalidad de los términos de interés para el programador.

La explicación de cada palabra o idea es breve y no muy detallada. Al final de algunas definiciones se dan referencias a otras para que el lector las consulte.

MSX de la A a la Z es una obra bastante completa como pequeño manual de consulta, ya que el abanico de conceptos que maneja es amplio. Sin embargo, dada la magnitud de la obra, se comprende que el espacio dedicado a cada palabra sea pequeño. No es en definitiva un libro para aprender, sino para recordar ideas y referencias olvidadas o poco claras, o bien para iniciar al usuario sin ningún conocimiento previo en las ideas básicas.